ICS 25.040
N 10

中华人民共和国国家标准

GB/T 20438.3—XXXX/IEC 61508-3:2010

电气/电子/可编程电子安全相关系统的功能安全　第3部分：软件要求

Functional safety of electrical/electronic/programmable electronic safety-related systems Part 3: Software requirements

(IEC 61508-3:2010, IDT)

(送审稿)

XXXX－XX－XX 发布　　XXXX－XX－XX 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

本标准等同采用国际标准IEC61508：2010《电气/电子/可编程电子安全相关系统的功能安全》（英文版）。

本标准由下列7部分构成：

——第 1 部分：一般要求

——第 2 部分：电气/电子/可编程电子安全相关系统的要求

——第 3 部分：软件要求

——第 4 部分：定义和缩略语

——第 5 部分：确定安全完整性等级的方法示例

——第 6 部分：第 2 部分和第 3 部分的应用指南

——第 7 部分：技术和措施概述

本标准的本部分等同采用国际标准IEC61508-3：2010《电气/电子/可编程电子安全相关系统的功能安全　第3部分：软件要求》（英文版）。

本部分附录A、附录B为规范性附录。

与本标准中规范性引用的国际文件有一致性对应关系的我国文件如下：

——GB/T 16499-2008　安全出版物的编写及基础安全出版物和多专业共用安全出版物的应用导则（IEC Guide 104:1997,NEQ）

——GB/T 20000.4-2003　标准化工作指南 第 4 部分:标准中涉及安全的内容(ISO/IEC Guide 51:1999,MOD)

本部分与IEC 61508-3：2010在技术内容上没有差异，为便于使用本标准作了下列编辑性修改：

a)　凡有“IEC61508”的地方改为“GB/T20438”；

b)　本“国际标准”一词改为“本标准”；

本标准由中国机械工业联合会提出。

本标准由全国工业过程测量和控制标准化技术委员会（SAC/TC124）归口。

本标准由.....单位负责起草。

本标准主要起草人：

引　言

由电气和电子组件构成的系统，多年来在许多应用领域中执行其安全功能。以计算机为基础的系统（一般指可编程电子系统)在其应用领域中用于执行非安全功能，并且也越来越多地用于执行安全功能。如果要使计算机系统技术有效和安全的使用，有关决策者在安全方面有充足的指导并据此做出决定是十分必要的 。

本标准针对由电气和/或电子和/或可编程电子组件（E/E/PE）构成的、起安全作用的系统整体安全生命周期的所有活动，提出了一个通用的方法。采用统一的方法的目的是为了针对以电为基础的安全相关系统提出一种一致的、合理的技术方针。主要目标是促进基于GB/T20438系列标准的产品和应用领域国家标准的制定。

注1：在参考目录中给出了基于 GB/T20438 系列标准的产品和应用领域国家标准的例子（见参考文献[1]，[2]，[3]）。

在许多情况下，可用多种基于不同技术的防护系统来保证安全（如机械的、液压的、气动的、电气的、电子的、可编程电子的等等）。因而必须考虑各类安全策略，不仅要考虑单个系统中的所有组件（如传感器、控制器、执行器等）问题，还应考虑不同安全系统组合后的问题。因此当本标准在关注电气/电子/可编程电子（E/E/PE）安全相关系统的同时，也提供了一个框架，在这个框架内，基于其它技术的安全相关系统也可被考虑进去。

在各种应用领域里，存在着许多潜在的危险和风险，包含的复杂性也各不相同，从而需应用不同的E/E/PE安全相关系统。对每个特定的应用，应根据应用的不同而确定所需的安全措施。本标准仅是使这些措施在未来的产品设计、生产和应用中保证国家标准和已经存在的这些标准的修订过程中的规范化。

本标准

——考虑了当使用 E/E/PE 系统执行安全功能时，所涉及到的整体安全生命周期、E/E/PE 系统安全生命周期以及软件安全生命周期的各阶段（如初始构思，整体设计、实现、运行和维护到停用）；

——针对飞速发展的技术，建立一个足够健壮而广泛的能满足今后发展需要的框架；

——使涉及 E/E/PE 安全相关系统的产品和应用领域的国家标准得以制定；在本标准的框架下，产品和应用领域的国家标准的制定在应用领域和交叉应用领域应具有高度一致性（如基本原理，术语等）；这将既具有安全性又具有经济效益；

——为达到 E/E/PE 安全相关系统所需的功能安全，提供了编制安全要求规范的方法；

——采用了一种可确定安全完整性要求的基于风险的方案；

——引入安全完整性等级，用于规定 E/E/PE 安全相关系统所要执行的安全功能的目标安全完整性等级；

注2：本标准没有规定每个安全功能的安全完整性等级的要求，也没有规定如何确定安全完整性等级。而是提供了一种基于风险概念的框架和技术范例。

——建立了 E/E/PE 安全相关系统的目标失效量，这些量都同安全完整性等级相联系；

——建立了危险失效模式中目标失效量的一个下限，此下限是对单一 E/E/PE 安全相关系统的要求。

这些E/E/PE安全相关系统运行在：

——低要求操作模式下，下限设定成要求时危险失效平均概率为 10^{-5}；

——高要求操作模式或者连续操作模式下，下限设定成危险失效平均频率为 10^{-9}/h；

注3：单一 E/E/PE 安全相关系统不一定是单通道架构。

注4：对于非复杂系统，通过安全相关系统的设计实现更优目标安全完整性是可能的。但对于相对复杂（例如可编程电子安全相关系统）的系统，这些限值代表了目前能够达到的水平。

——基于工业实践中获取的经验和判断，设定了避免和控制系统性故障的要求。即使发生系统性故障的可能性一般不能量化，但本标准允许为一个特定的安全功能做出声明，即如果标准中的所有要求都满足，认为与安全功能相关的目标失效量已达到；

——引入了系统性能力，该能力适用于一个组件关于系统性安全完整性满足规定的安全完整性等级要求的置信度。

采用多种原理、技术和措施以达到E/E/PE安全相关系统的功能安全，但没有明确的使用失效-安全的概念。然而，失效-安全的概念和本质安全原则是可以应用的，并且如果标准中相关条款的要求都满足，采用这些概念是可以接受的。

电气/电子/可编程电子安全相关系统的功能安全
第3部分：软件要求

1 范围

1.1 GB/T20438的本部分：

a) 应建立在充分理解GB/T20438.1和GB/T20438.2的基础上使用；

b) 适用于在GB/T20438.1和GB/T20438.2范围内构成安全相关系统的一部分或用于开发安全相关系统的任何软件。这种软件定义为安全相关软件(安全相关软件包括操作系统、系统软件、通信网络中的软件、人机界面功能、固件以及应用软件)；

c) 提供适用于在GB/T20438.1和GB/T20438.2范围内开发和配置安全相关系统的支持工具的特定要求；

d) 要求规定软件安全功能和软件系统性能力；

注1：如果这一要求作为电气/电子/可编程电子安全相关系统规范（见GB/T20438.2中7.2）的一部分已提出，则在此处不需重复。

注2：规定软件安全功能和软件系统性能力是一个反复的过程，见图3和图6。

注3：文档结构要求见GB/T20438.1的第5章和附录A。文档结构应考虑公司规程和特殊应用领域的工作实际情况。

注4：关于术语“系统性能力”的定义见GB/T20438.4的3.5.9。

e) 建立安全相关软件设计开发过程中（软件安全生命周期模型）对安全生命周期各阶段和需开展活动的要求。这些要求包括根据系统性能力分级的、在软件中用于避免和控制故障及失效的措施和技术的应用。

f) 对系统安全确认软件方面相关的信息提出了要求，这些信息将传递给执行E/E/PE系统集成的机构。

g) 对操作和维护E/E/PE安全相关系统的用户所需的软件有关的信息和规程的准备提出要求。

h) 对修改安全相关软件的机构提出要求。

i) 结合GB/T20438.1和GB/T20438.2，提出对支持工具的要求如设计开发工具、语言翻译器、测试和调试工具、配置管理工具。

注5：图5表示了GB/T20438.2和GB/T20438.3之间的关系。

j) 不适用于符合GB9706系列的医疗设备。

1.2 GB/T20438.1 、GB/T20438.2、GB/T20438.3和GB/T20438.4是基础的安全标准，虽然它不适用于低复杂的E/E/PE安全相关系统（见GB/T20438.4的3.4.3），但作为基础安全标准，各技术委员会可以在GB/T16499和GB/T20000.4的指导下制订相关标准时使用。GB/T20438.1、GB/T20438.2、GB/T20438.3和GB/T20438.4也可作为独立标准来使用。本标准的横向安全功能不适用于在GB9706系列指导下的医疗设备。

1.3 各技术委员会的责任之一，是在其标准的起草工作中尽可能使用基础的安全标准。在本部分中，本基础安全标准中的要求、测试方法或测试条件只有在这些技术委员会起草的标准中已明确引用或包含时适用。

1.4 图1表示了GB/T20438系列的整体框架，同时明确了在实现E/E/PE安全相关系统功能安全中本部分的作用。

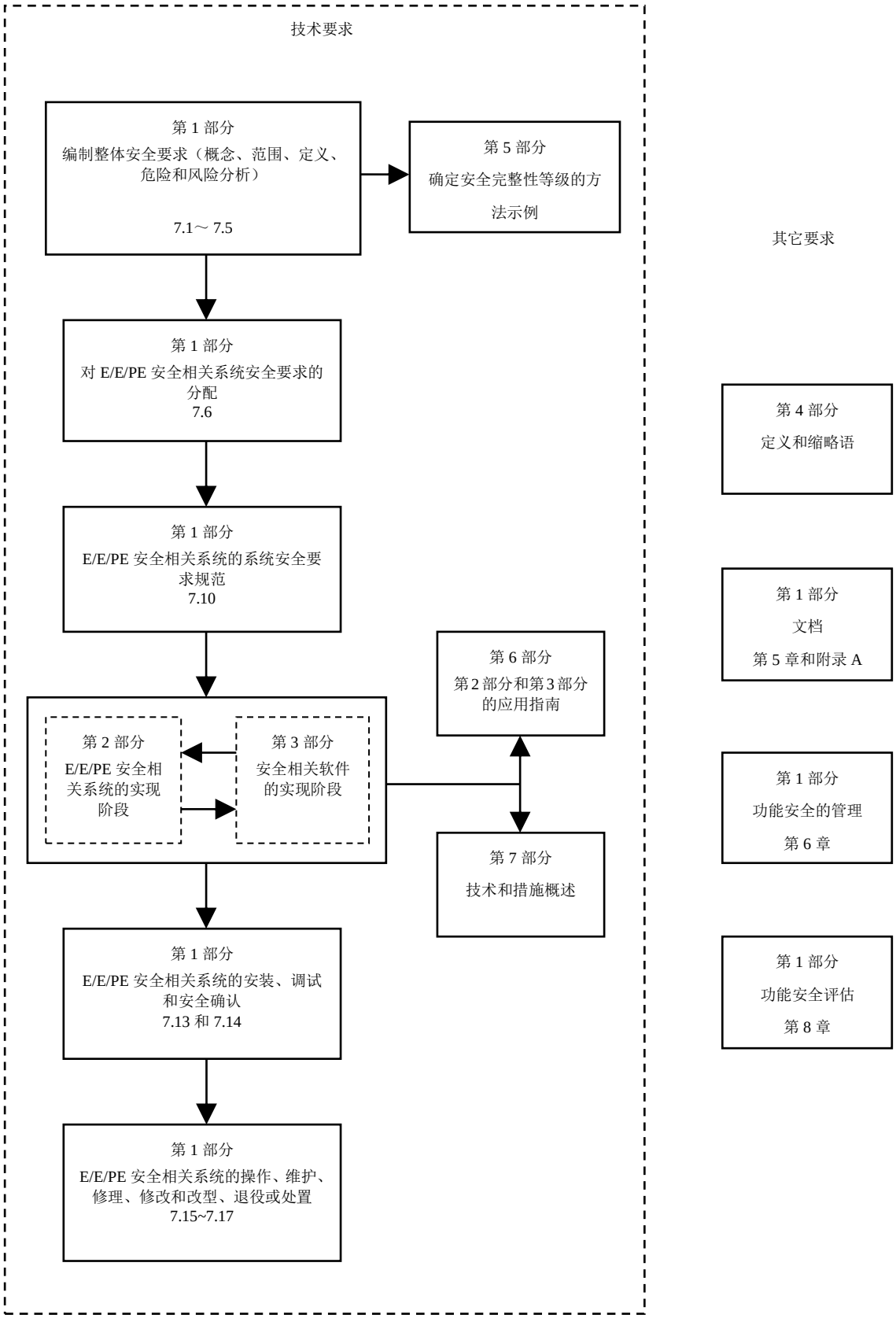

图1　GB/T20438 系列的整体框架

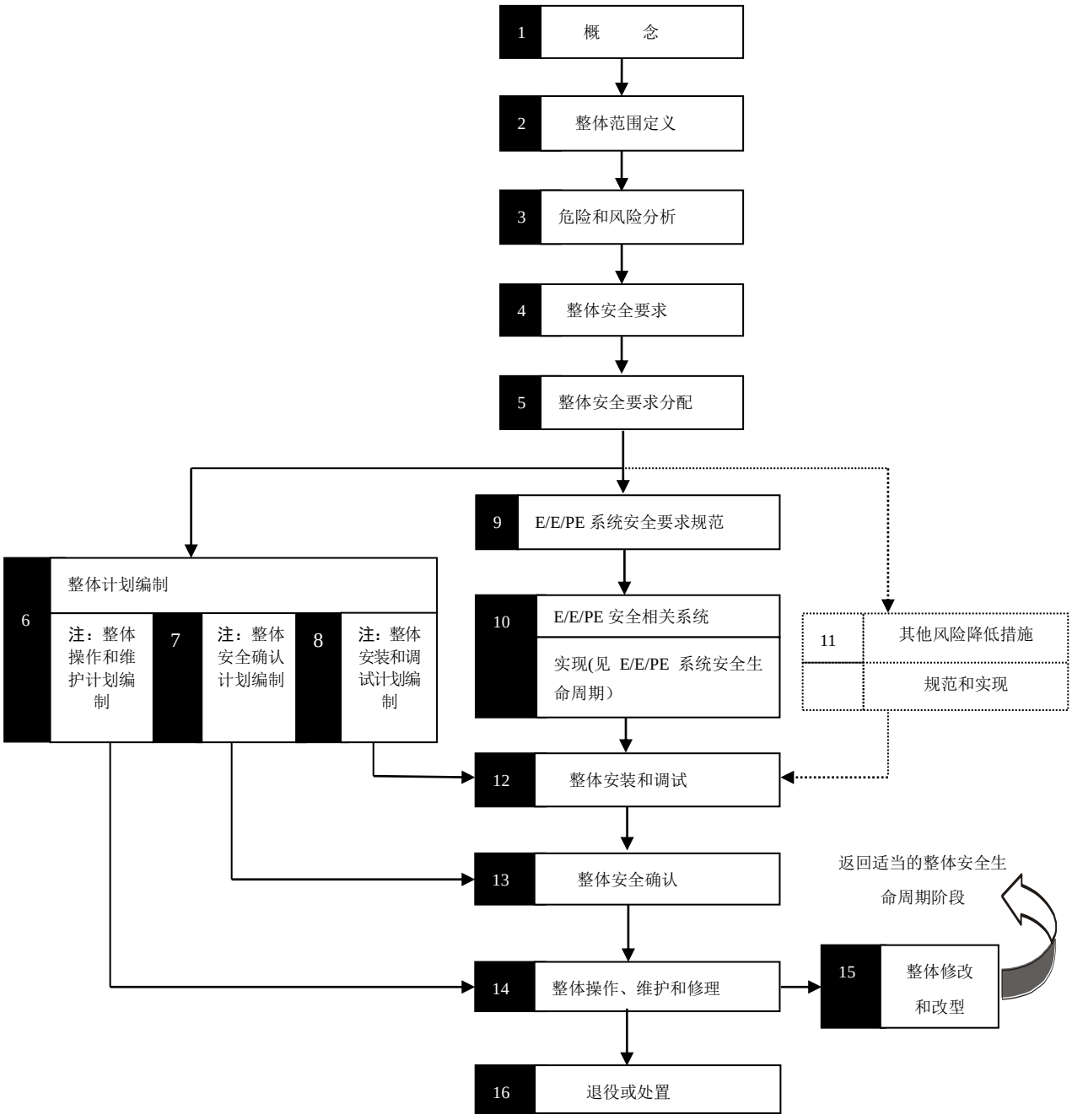
1 概 念
2 整体范围定义
3 危险和风险分析
4 整体安全要求
5 整体安全要求分配
9 E/E/PE 系统安全要求规范
6 整体计划编制
注：整体操作和维护计划编制
7 注：整体安全确认计划编制
8 注：整体安装和调试计划编制
10 E/E/PE 安全相关系统
实现(见 E/E/PE 系统安全生命周期)
11 其他风险降低措施
规范和实现
12 整体安装和调试
13 整体安全确认
返回适当的整体安全生命周期阶段
14 整体操作、维护和修理
15 整体修改和改型
16 退役或处置

图2 整体安全生命周期

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T20438.1-XXXX 电气/电子/可编程电子安全相关系统的功能安全 第1部分：一般要求（IEC 61508-1：2010，IDT）

GB/T20438.2-XXXX 电气/电子/可编程电子安全相关系统的功能安全 第2部分：电气/电子/可编程电子安全相关系统的要求(IEC 61508-2：2010，IDT)

GB/T20438.4-XXXX 电气/电子/可编程电子安全相关系统的功能安全 第4部分：定义及缩略语（ IEC 61508-4：2010，IDT）

GB/T20000.4：1999 安全部分--标准中包含安全内容的指南

IEC Guide 104:1997 安全出版物的编写及基础安全出版物和多专业共用安全出版物的应用导则（The preparation of safety publications and the use of basic safety）
publications and group safety publications

IEC/ISO Guide 51:1999 标准化工作指南 第4部分:标准中涉及安全的内容(Safety aspects - Guidelines for their inclusion in standards)

3 定义和缩略语

本部分采用GB/T20438.4-XXXX中规定的定义和缩略语。

4 标准的符合性

本部分对GB/T20438标准的符合性要求，详见GB/T20438.1-XXXX的第4章。

5 文档

本部分对文档的要求，详见GB/T20438.1-XXXX的第5章。

6 安全相关软件管理的附加要求

6.1 目的

见GB/T20438.1中6.1

6.2 要求

6.2.1 见 GB/T20438.1 中 6.2，以下为附加要求。

6.2.2 功能安全计划应规定由 E/E/PE 安全相关系统执行安全功能的安全完整性等级所需的软件采购、开发、集成、验证、确认和修改的策略。

注：该方法的理念是：考虑到E/E/PE安全相关系统执行每个安全功能所要求的安全完整性等级，用功能安全计划作为对本标准进行定制使用的手段。

6.2.3 软件配置管理

a) 应在整个软件安全生命周期中使用管理和技术控制以管理软件变更，从而保证安全相关软件的规定要求始终能得到满足；

b) 应确保所有必须的操作已被执行以证明达到了所需的软件系统性能力；

c) 应准确地和带唯一标识地维护满足 E/E/PE 安全相关系统性安全完整性要求所必须的所有配置项。

配置项至少包括：安全分析和要求、软件规范和设计文档、软件源代码模块、测试计划和结果、验证文档、将要被纳入E/E/PE安全相关系统的已有软件组件和软件包；所有用于创建、测试或执行E/E/PE安全相关系统软件的工具和开发环境；

d) 应采用变更控制规程：

——防止非授权的修改；归档修改请求；

——分析建议修改的影响以批准或拒绝请求；

——归档所有准许修改的细节和授权；

——在软件开发阶段中的适当点建立配置基线，并归档基线的（部分）集成测试；

——确保所有软件基线的构成和建立（包括早期基线的重建）；

注1：为指导、加强管理和技术控制的使用，有必要进行管理决定和授权。

注2：一种极端情况，影响分析可能包括一次非正式的评估；另一种极端情况，影响分析可能包括针对所有可能被不恰当地理解或执行的建议变更的潜在不利影响进行一次严格正式分析。见 GB/T20438.7 影响分析指南。

e) 确保使用适当方法，正确地加载有效软件组件和数据到运行时系统;

注3：本条款可能包括对特定目标系统以及通用系统的考虑。非应用软件可能需要一种安全的加载方法，如：固件。.

f) 应归档下列信息，以用于随后的功能安全审核：配置状态、发布状态、对所有修改的论证（考虑影响分析)、批准和修改的详细情况；

g) 应正式归档安全相关软件的发布。软件的主要备份和所有有关文档和服务的数据版本在已发布软件的操作生命周期内应被保存，以允许维护和修改。

注4：对于配置管理的更详细的信息，见 GB/T 20438.7。

7 软件安全生命周期要求

7.1 概述

7.1.1 目的

本章的目的是将软件开发纳入到已定义的各阶段和活动中（见表1、图3至图6）。

7.1.2 要求

7.1.2.1 软件开发的安全生命周期应在根据GB/T20438.1第6章的安全计划编制期间进行挑选和规定。

7.1.2.2 只要本章的所有目的和要求都得到满足，可以使用任何的软件生命周期模型。

7.1.2.3 软件安全生命周期的每个阶段应分成若干基本活动，每个阶段应规定范围、输入和输出。

注：见图3、图4和表1。

7.1.2.4 只要软件安全生命周期满足表 1 要求，允许根据项目的安全完整性和复杂性对 V 模型（见图6）进行裁剪。

注1：满足本章要求的软件安全生命周期模型可按照项目或组织的需要适当定制。表 1 中生命周期各阶段的全部列表更适用新开发的大型系统。对于小的系统，例如将软件系统设计和架构设计合并也是合适的。

注2：数据驱动系统特性(例如：全可变/有限可变编程语言、数据配置的程度)见附录 G，这些特性在定制软件安全生命周期时可能是相关的。

7.1.2.5 软件安全生命周期的定制应基于功能安全进行论证。

7.1.2.6 质量和安全保证规程应该集成到安全生命周期活动中。

7.1.2.7 对生命周期的每个阶段，应使用适当的技术和措施。附录 A 和 B 提供了选择技术和措施的指南，并参考 GB/T20438.6 和 GB/T20438.7。GB/T20438.6 和 GB/T20438.7 给出了达到系统性安全完整性要求属性的特定技术推荐。仅从这些推荐中选择技术不能保证就实现了要求的安全完整性。

注：成功实现系统性安全完整性依赖于在选择技术时考虑如下因素：

——为整个开发周期选择的方法、语言和工具的一致性和互补性；

——开发者是否完全理解使用方法、语言和工具；

——在开发期间，方法、语言和工具是否对遇到的特定问题有很好的适用性。

7.1.2.8 软件安全生命周期中的活动结果应归档（见第 5 章）。

注：GB/T20438.1第5章考虑了安全生命周期各阶段输出的文档。在E/E/PE安全相关系统的开发中，某些安全生命周期阶段的输出文档可能是单独的，而一些阶段的输出文档可能是合并的。本质的要求是安全生命周期阶段输出应与预期目的相匹配。

7.1.2.9 如果在软件安全生命周期的任一阶段，要求对早期生命周期阶段改变时，那么要进行影响分析以确定：（1）哪个软件模块受到影响（2）早期安全生命周期的哪些活动应该重复。

注：一种极端情况，影响分析可能包括一次非正式的评估；另一种极端情况，影响分析可能包括针对所有可能被不恰当地理解或执行的建议变更的潜在不利影响进行一次严格正式分析。见GB/T20438.7影响分析指南。

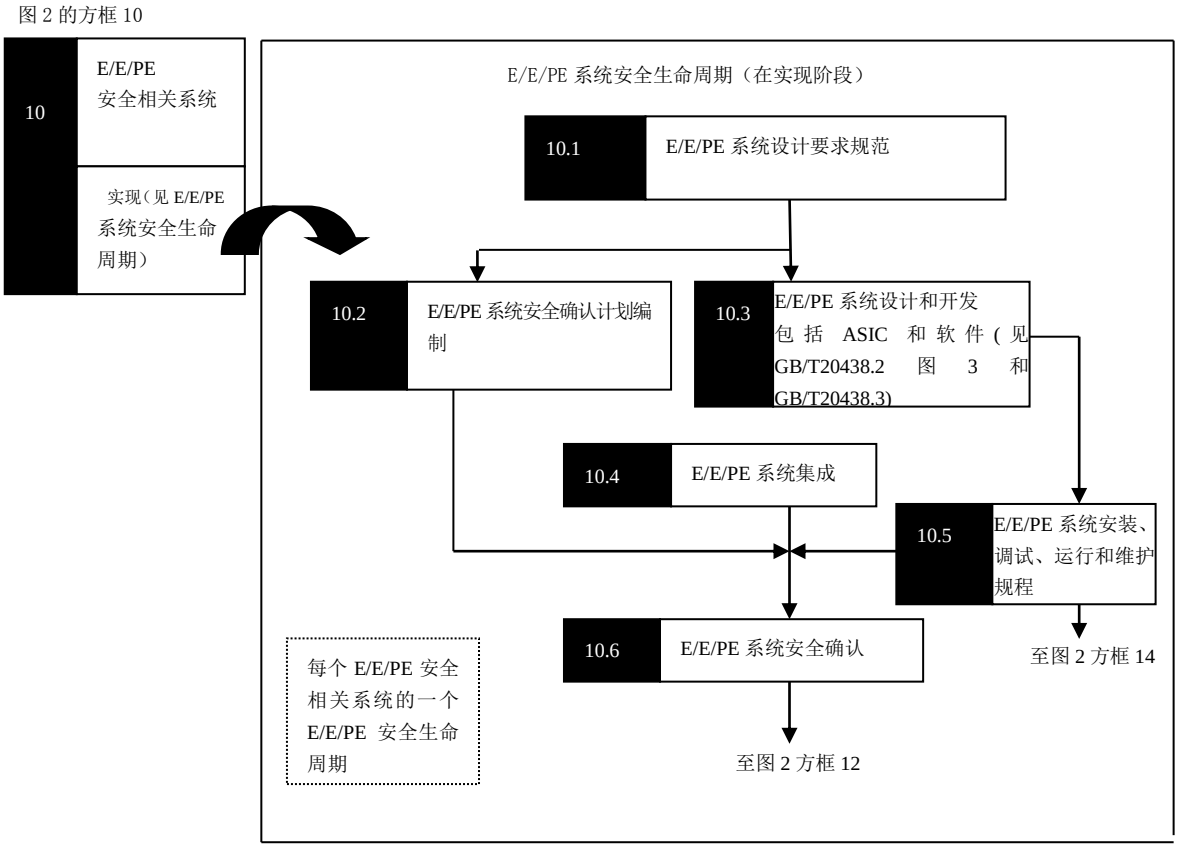

图3 E/E/PE 系统安全生命周期（在实现阶段）

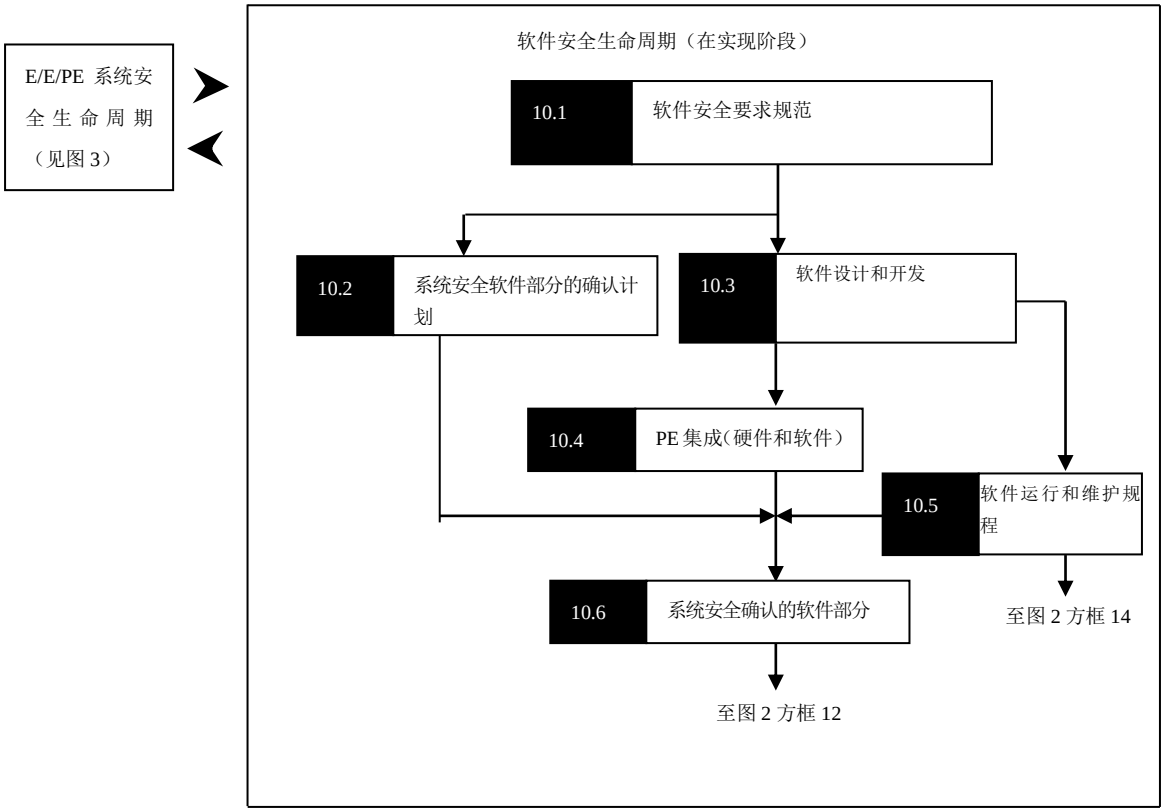

图4　软件安全生命周期（在实现阶段）

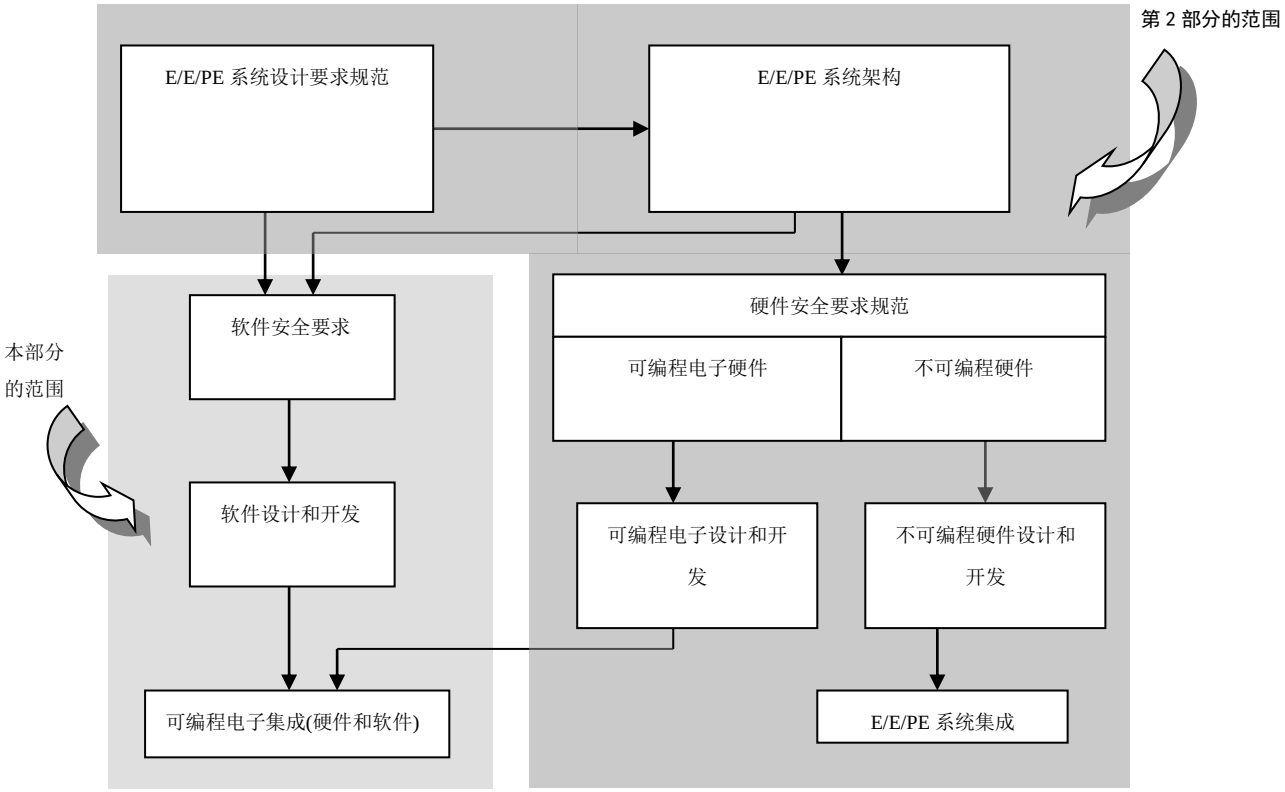

图5　GB/T20438.2 和 GB/T20438.3 的范围和关系

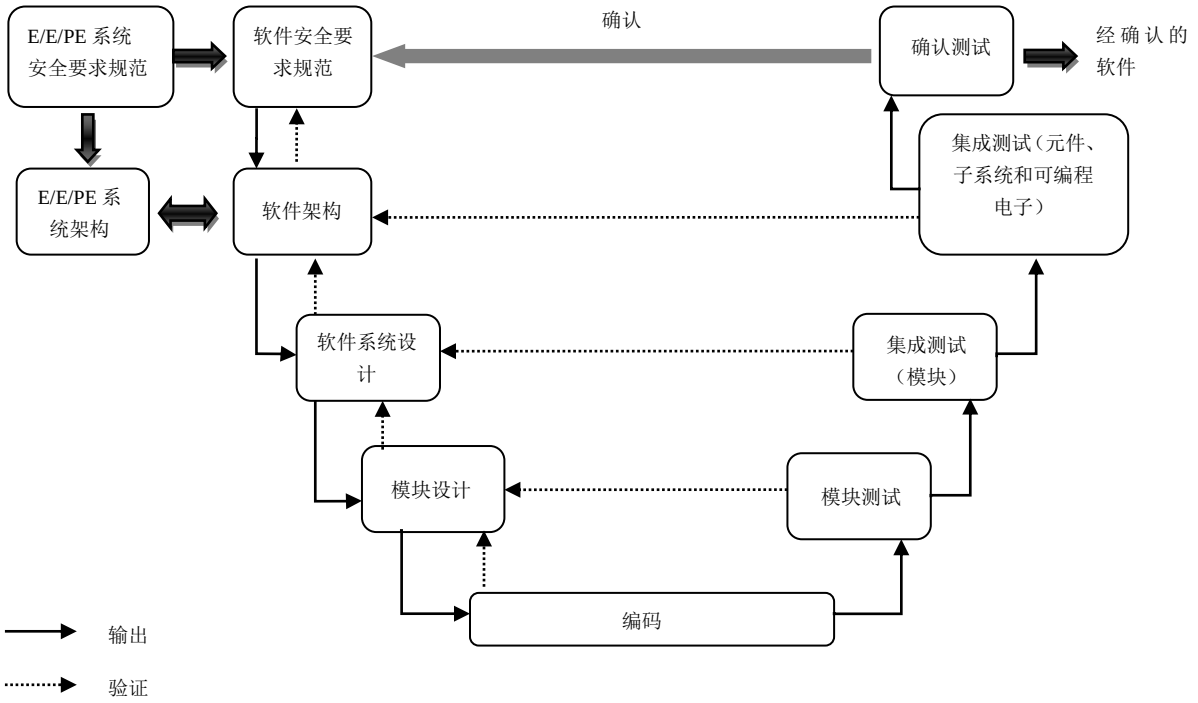

图6　软件系统性能力和开发生命周期（V 模型）

表1　软件安全生命周期：概述

安全生命周期阶段		目的	范围	要求所在的条款	输入（要求的信息）	输出(产生的信息)
图 4 中的方框号	标题					
10.1	软件安全要求规范	根据软件安全功能要求和软件系统性能力要求规定软件安全要求； 对每个需要执行一定安全功能的 E/E/PE 安全相关系统规定软件安全功能的要求； 规定每一个 E/E/PE 安全相关系统对于软件系统性能力的要求，以保证获得这一 E/E/PE 系统分配的每一安全功能需达到的安全完整性等级	PE 系统； 软件系统	7.2.2	在分配阶段生成的 E/E/PE 安全要求规范（见 GB/T20438.1） E/E/PE 系统安全要求规范（来自于 GB/T20438.2）	软件安全要求规范
10.2	系统安全软件部分的确认计划	编制系统安全的软件确认计划	PE 系统； 软件系统	7.3.2	软件安全要求规范	系统安全软件部分的确认计划
10.3	软件设计和开发	架构： 创建软件架构，满足不同的安全完整性等级中对软件安全规定要求； 评价 E/E/PE 安全相关系统硬件架构对软件的要求，包括 E/E/PE 中软件/硬件相互作用对受控设备安全性的影响。	PE 系统； 软件系统	7.4.3	软件安全要求规范； E/E/PE 系统硬件架构设计（来自于 GB/T20438.2）	软件架构设计； 软件架构集成测试规范； 软件/可编程电子集成测试规范（在 GB/T20438.2 中也做了要求）
10.3	软件设计和开发	支持工具和编程语言： 在用于辅助验证、确认、评估和修改的软件整个生命周期中，根据要求的安全完整性等级选择合适的工具集，包括语言和编译器、运行时系统接口、用户界面、数据格式和表现形式	PE 系统； 软件系统； 支持工具； 编程语言。	7.4.4	软件安全要求规范； 软件架构设计	支持工具和编码标准； 开发工具的选择

表 1 （续）

安全生命周期阶段		目的	范围	要求所在的条款	输入（要求的信息）	输出（产生的信息）
图 4 中的方框号	标题					
10.3	软件设计和开发	详细设计和开发（软件系统设计）： 设计和实现软件，以满足要求的安全完整性等级对安全相关软件规定的要求，这种软件可分析、可验证并能被安全地修改	软件架构设计的主要组件和子系统	7.4.5	软件架构设计； 支持工具和编码标准；	软件系统设计规范； 软件系统集成测试规范
10.3	软件设计和开发	详细设计和开发（单个软件模块设计）： 设计和实现软件，以满足要求的安全完整性等级对安全相关软件规定的要求，这种软件可分析、可验证并能被安全地修改	软件系统设计	7.4.5	软件系统设计规范； 支持工具和编码标准	软件模块设计规范； 软件模块测试规范
10.3	软件设计和开发	详细代码实现： 设计和实现软件，以满足要求的安全完整性等级对安全相关软件规定的要求，这种软件可分析、可验证并能被安全地修改	单个软件模块	7.4.6	软件模块设计规范； 支持工具和编码标准	源代码清单； 代码复审报告
10.3	软件设计和开发	软件模块测试： 验证对安全相关软件的要求已实现（根据要求的软件安全功能和软件系统性能力）。 说明每一软件模块以实现其预定的功能，而不实现非预定的功能。 在适用的范围内，确保用数据配置 PE 系统能满足软件系统性能力的特定要求	软件模块	7.4.7	软件模块测试规范； 源代码清单； 代码复审报告	软件模块测试结果； 经验证和测试的软件模块
10.3	软件设计和开发	软件集成测试： 验证对安全相关软件的要求已实现（根据要求的软件安全功能和软件系统性能力）。 说明所有软件模块、组件和子系统相互正确作用以实现其预定的功能，而不实现非预定的功能。 在适用的范围内，确保用数据配置 PE 系统能满足软件系统性能力的特定要求	软件架构； 软件系统	7.4.8	软件系统集成测试规范	软件系统集成测试结果； 经验证和测试的软件系统；

表 1 （续）

安全生命周期阶段		目的	范围	要求所在的条款	输入（要求的信息）	输出（产生的信息）
图 3 中的方框号	标题					
10.4	可编程电子集成（硬件和软件）	在目标可编程电子硬件上集成软件； 将软件和硬件结合到安全相关的可编程电子上以保证其兼容性和满足预定安全完整性等级的要求。	可编程电子硬件； 集成的软件	7.5.2	软件架构集成测试规范； 软件/PE 集成测试规范（在 GB/T20438.2 中也做了要求）； 集成的可编程电子	软件架构集成测试结果； 可编程电子集成测试结果； 经验证和测试的集成可编程电子
10.5	软件运行和修改规程	提供软件有关的信息和规程以保持运行 和修改阶段中 E/E/PE 安全相关系统的功能安全。	同上	7.6.2	与上面所有内容相关的	软件运行和修改规程
10.6	系统安全确认的软件方面	保证集成的系统符合安全相关软件在预定安全完整性等级下的规定要求	同上	7.7.2	系统安全的软件确认计划	软件安全确认结果； 已确认软件；
——	软件修改	修正、增强或调整确认软件以保证维持所要求的软件系统性能力	同上	7.8.2	软件修改规程； 软件修改请求	软件修改影响分析结果； 软件修改日志
——	软件验证	测试和评价给定软件安全生命周期阶段的输出，以保证当输入该阶段时提供的输出与标准的正确性和一致性。	根据阶段	7.9.2	适当的验证计划（根据阶段）	适当的验证报告（根据阶段）
——	软件功能安全评估	调查并对 E/E/PE 安全相关系统所实现的功能安全的软件方面作出判断	所有以上阶段	8	软件功能安全评估计划	软件功能安全评估报告

7.2 软件安全要求规范

注：这一阶段是图4中的方框10.1。

7.2.1 目的

7.2.1.1 本章的第一个目的是规定对安全相关软件的要求，包括软件安全功能的要求和软件系统性能力的要求。

7.2.1.2 本章的第二个目的是针对每个执行要求的安全功能的 E/E/PE 安全相关系统，规定软件安全功能的要求。

7.2.1.3 本章的第三个目的是针对每个 E/E/PE 安全相关系统要求达到的安全完整性等级，规定软件系统性能力的要求。

7.2.2 要求

注1：这些要求大多情况下可由通用嵌入式软件和特定应用软件共同满足。要求两者结合来提供满足下列条款的特性。两者之间的精确划分依据所选择的软件架构（见 7.4.3）。

注2：为了选择合适的技术和措施（见附录 A 和 B）来执行本章的要求，宜考虑如下的软件安全要求规范的属性（属性的解释指南见附录 C 和 GB/T20438.7 附录 F 非正式定义）：

——需要由软件来确保安全方面的完整性；

——需要由软件来确保安全方面的正确性；

——没有规范本身的错误，包括：语义含糊；

——安全要求的可理解性；

——没有非安全功能对由软件来确保安全的不利干扰；

——为验证和确认提供基础的能力。

注3：由软件提供的安全应是一组安全功能与相应安全完整性要求，上述内容是通过 E/E/PE 系统的设计分配给软件功能的。(系统安全要求的完整集是一个更大的集，也包括不依靠软件的安全功能)。软件安全要求规范的完整性非常依赖早期的系统生命周期阶段的有效性。

7.2.2.1 如果安全相关软件的要求已在 E/E/PE 安全相关系统的要求中规定（见 GB/T20438.2 中第 7 章），则在软件安全要求规范中不必重复。

7.2.2.2 安全相关软件要求规范应由 E/E/PE 安全相关系统规定的安全要求（见 GB/T20438.2 第 7 章）和任何安全计划的要求（见第 6 章）中得出，软件开发人员应能获取这些信息。

注1：这一要求并不意味着 E/E/PE 系统开发人员和软件开发人员之间没有反复（GB/T20438.2 和 GB/T20438.3），当安全相关软件要求和软件架构变得更加精确时，将会对 E/E/PE 系统硬件架构产生影响，因此软件和硬件开发人员之间的密切合作就变得非常必要了，见图 5。

注2：软件设计包含已有重复使用软件时，原软件的开发可能没有考虑当今系统要求规范。对已有软件满足软件安全要求规范的要求见 7.4.2.12。

7.2.2.3 安全相关软件要求规范应足够细致以使设计和实施能达到要求的安全完整性(包括独立性要求，见 GB/T20438.2 中 7.4.3)，并允许执行功能安全的评估。

注：规范的细致程度可根据应用的复杂程度确定。一个适当的功能行为规范可包括精度、时序和性能、容量、健壮性、超载限度和特定应用特征属性的要求。

7.2.2.4 为了实现独立性，要执行适当的共因失效分析。当识别可信的失效机制时，应采取有效的防御措施。

注：软件独立性实现技术见附录F。

7.2.2.5 软件的开发人员应评估 7.2.2.2 中的信息以确保对要求充分规定，应特别考虑以下内容：

a) 安全功能；

b) 系统配置或架构；

c) 硬件安全完整性要求（可编程电子、传感器和执行器）；

d) 软件系统性能力要求；

e) 容量和响应时间；

f) 设备和操作人员界面，包括合理可预见的错误。

注：应考虑与已有应用相兼容。

7.2.2.6 如果没有在 E/E/PE 安全相关系统的规定安全要求中充分定义，EUC、E/E/PE 系统以及连接到 E/E/PE 系统的其他设备或系统的所有相关运行模式应在规定的安全相关软件要求中详细说明。

7.2.2.7 软件安全要求规范应该规定和归档软硬件之间的所有安全相关的或有关的约束。

7.2.2.8 在 E/E/PE 硬件架构设计要求的范围内，并考虑到可能增加的复杂性，软件安全要求规范应该考虑下面内容：

a) 软件自监视（例子见 GB/T20438.7）；
b) 可编程电子硬件、传感器和执行器的监视；
c) 在系统运行时，安全功能的周期测试；
d) 当 EUC 运行时，使安全功能可测试；
e) 为了满足 E/E/PE 安全相关系统性安全完整性要求，执行所有检验测试和自诊断测试的软件功能。

注：对于上述考虑导致的增加的复杂性可能要求更改架构。

7.2.2.9 当要求 E/E/PE 安全相关系统执行非安全功能时，安全相关软件的规定要求应明确识别出非安全功能要求。

注：安全功能和非安全功能间无干扰的要求见7.4.2.8和7.4.2.9。

7.2.2.10 软件安全要求规范应该表达要求的产品安全属性，而不是项目的安全属性，这些在安全计划（见 GB/T20438.1 第 6 章）中已经覆盖。参考 7.2.2.1 到 7.2.2.9，应适当地规定如下内容：

a) 对下列软件安全功能的要求：
   1) 使 EUC 实现或维持安全状态的功能；
   2) 在可编程电子硬件中，检测、报告和管理故障的相关功能；
   3) 检测、报告和管理传感器和执行器故障的相关功能；
   4) 在软件自身内部，检测、报告和管理故障的相关功能（软件自监视）；
   5) 安全功能在线（即在预期的运行环境中）周期测试的相关功能；
   6) 安全功能离线（即 EUC 处于不依赖于其安全功能的环境中）周期测试的相关功能；
   7) 允许安全修改 PE 系统的功能；
   8) 与非安全相关功能的接口；
   9) 容量和响应时间性能；
   10) 软件和 PE 系统之间的接口；

注1：包括离线和在线的编程设备。

   11) 安全相关通信（见 GB/T20438.2 的 7.4.11）。

b) 对软件系统性能力的要求：
   1) 在上述 a) 中每个功能的安全完整性等级；

注2：关于分配安全完整性给软件组件的信息见 GB/T20438.5 附录 A。

   2) 功能间的独立性要求。

7.2.2.11 如果软件安全要求由配置数据表示或执行，数据应：

a) 与系统安全要求一致；
b) 按照允许范围和经授权的一组运行参数来表示；
c) 以关联软件（例如执行顺序、运行时间、数据结构等）兼容的方式定义。

注1：这个对应用数据的要求特别针对数据驱动的应用。有下面的特性：源码已经存在，开发活动的主要目的是保证配置数据正确表达了应用要求的行为。在数据项之间可能有复杂的依赖关系，并且数据有效性可能随时间改变。

注2：数据驱动系统指南见附录 G。

7.2.2.12 如果数据定义软件和外部系统间的接口，在 GB/T20438.2 的 7.4.11 之外，应该考虑下面的性能特性：

a) 需要与数据定义一致；
b) 无效的、超出范围或不适时的值；
c) 响应时间和吞吐量，包括最大负载条件；
d) 最好和最坏情况的执行时间，和死锁；
e) 数据存储容量的上溢和下溢。

7.2.2.13 应保护运行参数，以防止：

a) 无效的、超出范围或不适时的值；
b) 非授权变更；
c) 损坏。

注1：应基于软件和非软件机制，考虑防止非授权变更。注意有效防止非授权的软件更改可能对安全产生不利的影响，比如当变更需要在快速和有压力条件下进行时。

注2：虽然人员可以是安全相关系统的一部分（见 GB/T20438.1 第 1 章），与 E/E/PE 安全相关系统设计相关的人的因素要求在本标准中没有详细考虑。然而，应适当考虑下面关于人的因素：

——操作员信息系统宜用图形化布局和操作员熟悉的术语。它应是清楚的，容易理解并且没有不必要的细节和/或方面；

——关于 EUC 向操作员显示的信息应该密切匹配 EUC 的物理布局；

——如果操作员面对几个显示内容，和/或可能的操作员动作允许产生互动，其后果不能一目了然，显示的信息应该自动包括一个显示或一个动作顺序的每种状态，信息应显示何种序列状态已经到达、何种操作可行和何种可能的后果可选择。

7.3 系统安全软件方面的确认计划

注1：本阶段对应图 4 中的方框 10.2。

注2：软件确认通常不能脱离它的底层硬件和系统环境。

7.3.1 目的

本章的目的是制定系统安全的安全相关软件部分的确认计划。

7.3.2 要求

7.3.2.1 应编制计划来规定规程上和技术上的步骤，用以证明软件满足其安全要求。

7.3.2.2 系统安全软件部分的确认计划应考虑：

a) 何时确认的细节；
b) 何人进行确认的细节；
c) EUC 相关运行模式的确定，包括：
   1) 使用的准备，包括设置和调整；
   2) 启动、教学、自动、手动、半自动、稳定状态运行；
   3) 重启、关机、维护；
   4) 合理可预见的异常状况和误操作。
d) 在开始调试前针对每个 EUC 运行模式需要进行确认的安全相关软件的确定；
e) 确认的技术策略（如分析方法、统计测试等）；
f) 根据 e)，用于证实每个安全功能符合规定的安全功能要求和规定的软件系统性能力要求的措施（技术）和规程；

g) 进行确认活动时要求的环境（如测试包括的校准工具和设备）；

h) 通过/失败的准则；

i) 评价确认结果的方针和规程，特别是失败时。

注：这些要求基于 GB/T20438.1 中 7.8 的一般要求。

7.3.2.3 确认应给出所选策略的基本原理。安全相关软件确认的技术策略应包括下列信息：

a) 选择手动或自动技术或两者均选；

b) 选择静态或动态技术或两者均选；

c) 选择分析或统计技术或两者均选；

d) 选择接受的准则是基于客观因素或专家判断还是两者均选。

7.3.2.4 如果安全完整性等级有要求（见 GB/T20438.1 第 8 章），则作为系统安全软件方面确认计划编制规程的一部分，系统安全软件方面确认计划的范围和内容应得到评估人员或评估方代表的认可。该规程还应说明在测试过程中评估人员是否需要出席。

7.3.2.5 完成软件确认通过/失败的准则应包括：

a) 要求的输入信号及其序列和值；

b) 预期的输出信号及其序列和值；

c) 其它可接受的准则，如存储使用、时序和数值允差。

7.4 软件设计和开发

注：这一阶段为图4中的方框10.3。

7.4.1 目的

7.4.1.1 本章的第一个目的是创建软件架构以满足要求的安全完整性等级中对安全相关软件规定的要求。

7.4.1.2 本章的第二个目的是评价由 E/E/PE 系统安全相关系统硬件架构对软件的要求，包括 E/E/PE 系统中软件和硬件相互作用对受控设备安全性的影响。

7.4.1.3 本章的第三个目的是为要求的安全完整性等级选择合适的工具集，包括语言和编译器、运行时系统接口、用户界面、数据格式及其表示法，在软件的整个安全生命周期中，这些工具用于辅助验证、确认、评估和修改。

7.4.1.4 本章的第四个目的是设计和实现软件，以满足要求的安全完整性等级对安全相关软件规定的要求，并使软件可分析、可验证并能被安全地修改。

7.4.1.5 本章的第五个目的是验证安全相关软件要求（根据要求的软件安全功能和软件系统性能力）已经实现。

7.4.1.6 本章的第六个目的是在适用的范围内，确保 PE 系统的数据配置满足软件系统性能力的规定要求。

7.4.2 一般要求

7.4.2.1 根据软件开发的性质，符合 7.4 的责任取决于安全相关编程环境的供方（如 PLC 供应商），或取决于该环境的用户（如应用软件开发者），或取决于两者。责任的划分应在安全计划编制过程中确定（见第 6 章）。

注：在系统和软件架构方面，实际责任的划分见7.4.3。

7.4.2.2 根据要求的安全完整性等级和安全功能特定技术要求，选择的设计方法应具有便于以下内容的特性：

a) 抽象化、模块化和控制复杂性的其它特性；

b) 表达以下内容:

1) 功能性;
2) 组件间的信息流;
3) 序列和时间相关的信息;
4) 时序约束;
5) 访问共享资源的并发性和同步性;
6) 数据结构及其属性;
7) 设计假设及其关联性;
8) 异常处理;
9) 设计假设（前提条件、后置条件、不变性）;
10) 注释。

c) 表达包括结构视图和行为视图在内的若干设计视图的能力;

d) 使开发者和其他需要了解设计的人员理解;

e) 验证和确认。

7.4.2.3 为了有助于这些属性在最终安全相关系统中的执行，在设计活动中应考虑可测试性和安全修改的能力。

注：例子包括机械和过程装置中的维护模式。

7.4.2.4 选择的设计方法应具有方便软件修改的特性。这些特性包括模块化、信息隐藏和封装。

注：见附录F.7。

7.4.2.5 设计表达方式应依据明确定义的或受限于明确定义了特征的符号表示法。

7.4.2.6 设计应尽可能将软件中的安全相关部分简单化。

7.4.2.7 软件设计应包括与要求的安全完整性等级相对应的控制流和数据流的自监视。在检测到失效时应采取适当的行动。

7.4.2.8 当软件同时实现安全功能和非安全功能时，所有的软件都将被认为是安全相关的，除非有足够的设计措施保证非安全功能失效不会对安全功能产生不利影响。

7.4.2.9 当软件执行不同安全完整性等级的安全功能时，所有的软件都被认为属于最高安全完整性等级，除非在设计中表明不同安全完整性等级的安全功能之间的充分独立性。应证明（1）独立性在空间域和时间域上均得到实现，或（2）任何对独立性的违背均得到控制。对独立性的论证应归档。

注：有关软件独立性的实现技术见附录F。

7.4.2.10 如果软件组件的系统性能力低于其提供的安全功能的安全完整性等级，组件应与其他组件组合使用，以使组合的系统性能力等于安全功能的安全完整性等级。

7.4.2.11 如果使用已知系统性能力的软件组件组合来执行安全功能，应对组件组合应用 GB/T 20438.2 的 7.4.3 中系统性能力要求。

注：要始终区分（1）由一个或多个组件支持的端到端安全功能和（2）每个支持组件的组件安全功能。如果两个组件组合以实现更高的系统性能力，成对组件中的每个组件都应该能防止/减轻危险事件，但成对组件不要求具有同样的组件安全功能，并且不要求每个组件能够单独提供组合所要求的全部安全功能性。

示例：一个电子引擎节流阀控制的端到端安全功能是“防止非要求加速”。端到端安全功能是通过两个处理器来实现。主处理器的组件安全功能是节流阀理想的要求/响应行为。另一个处理器的组件安全功能是一个多样性监视器（见 GB/T 20438.7 的 C.3.4），并在必要时应用紧急停止。两个处理器的组合实现了端到端安全功能“防止非要求加速”更高的置信度。

7.4.2.12 当一个已有的软件组件被复用，实现所有或部分安全功能时，对于系统性安全完整性，该组件应同时满足以下 a)和 b)：

a) 满足以下符合性路线之一：

——路线 1s：符合性开发。符合本标准的要求，以避免并控制软件中的系统性故障；

——路线 2s：经使用证明。提供组件已在使用中证明的证据。见 GB/T 20438.2 的 7.4.10 ；

——路线 3s：非符合性开发评估。符合 7.4.2.13。

注1：路线 1s、2s 和 3s 是 GB/T 20438.2 的 7.4.2.2 c）中的组件符合性路线在软件组件上的引用。在此处重复出现仅是为了方便，以减少引用 GB/T 20438.2。

注2：见 GB/T 20438.4 的 3.2.8。已有软件可能是商用产品，或是为以往的产品或系统由某组织开发的。已有软件可能是依照本标准的要求进行开发的，但也可能不是。

注3：对已有组件的要求适用于运行时库或一个解释器。

b) 提供安全手册（见 GB/T 20438.2 的附录 D 和本部分的附录 D）以给出足够精确和完整的组件描述，使得一个整体或部分依赖于已有软件组件的特定安全功能的完整性评估成为可能。

注4：安全手册可来源于组件供应商自己的文档和组件供应商开发流程的记录，或者由安全相关系统的开发者或第三方承担的额外具备资质的活动来创建或补充。有时主要由于法律条件（例如，版权或者知识产权），可能需要逆向工程来创建规范或设计文档来充分满足本章的要求。

注5：组件的论证可在安全计划编制期间进行（见第 6 章）。

7.4.2.13 为遵守路线 3s，一个已有软件组件应满足下列 a)到 i)的全部要求：

a) 组件的软件安全要求规范在其新应用中应归档，此文档应按照本部分为任何同等系统性能力的安全相关组件要求的同等精确程度建立。软件安全要求规范应覆盖组件在其新应用中适用的功能和安全行为及 7.2 中的规定。见表 A.1。

b) 软件组件的使用论证应提供足够的证据，表明在附录 C 的指导下考虑了相关条款（即 7.2.2、7.4.3、7.4.4、7.4.5、7.4.6、7.4.7、7.5.2、7.7.2、7.8.2、7.9.2 和第 8 章）规定的期望安全属性。

c) 组件设计应归档，并具备一定程度的精确性和充分性，以对其符合规范要求和符合要求的系统性能力提供证据。见 7.4.3、7.4.5 和 7.4.6，以及附录 A 的表 A.2 和表 A.4。

d) 7.4.2.13 a)和 7.4.2.13 b)要求的证据应覆盖硬件与软件的集成。见 7.5 和附录 A 的表 A.6。

e) 应有证据表明，组件已采用系统性方法进行验证与确认，该方法包括文档化的测试和对所有组件设计和代码的复审。见 7.4.7、7.4.8、7.5、7.7 和 7.9，以及附录 A 的表 A.5 到 A.7 和 A.9，还有附录 B 中的相关表格。

注1：正面的操作经验可用于满足黑盒测试和概率测试要求[见表 A.7 和 B.3]。

f) 如果软件组件提供安全相关系统中不需要的功能，那么应提供证据表明不需要的功能不会阻止 E/E/PE 系统满足其安全要求。

注2：满足此要求的方法包括：

——从构造中移除这些功能；

——禁止这些功能；

——适当的系统架构（如分块、封装、多样性、输出可信度校验）；

——广泛的测试。

g) 应有证据表明已经识别了软件组件的所有可信失效机制并采取了适当的缓解措施。

注3：适当的缓解措施包括：

——适当的系统构架（如分块、封装、多样性、输出可信度校验）；

——异常处理。

h) 组件使用计划应确定软件组件的配置、软件和硬件运行时环境和编译/连接系统的配置（如有必要）。

i) 使用组件的论证仅对其应用遵从组件符合项安全手册所做的假设时有效(见 GB/T20438.2 的附录 D 和本部分的附录 D）。

7.4.2.14 在适用的范围内，7.4.2 条应用于数据和数据生成语言。

注：关于数据驱动系统的指南见附录G。

a) 当 PE 系统由通过数据进行配置的已有功能组成来满足特定应用要求时，应用软件的设计应与应用的可配置度、已交付的既有功能和 PE 安全相关系统的复杂度相匹配。

b) 当 PE 系统的安全相关功能在很大程度上或主要取决于配置数据时，应使用适当的技术和措施以防止在配置数据的设计、生产、装载和修改期间引入故障，并确保配置数据正确地描述应用逻辑。

c) 数据结构规范应：

——与系统的功能要求一致，包括应用数据；

——完整；

——具有自身一致性；

——保护数据结构，防止改变和破坏。

d) 当 PE 系统由通过数据进行配置的已有功能组成来满足特定应用要求时，配置过程自身应相应归档。

7.4.3 软件架构设计的要求

注1：软件架构定义软件的主要组件和子系统，包括它们如何实现内部连接，如何实现所要求的属性，特别是安全完整性。还定义软件的整体行为、软件组件的接口和相互作用。主要软件组件包括操作系统、数据库、EUC 输入/输出子系统、通信子系统、应用程序、编程和诊断工具等。

注2：在某些工业领域中，软件架构可称作功能描述或功能设计规范（尽管这些文档也可包括硬件）。

注3：在某些用户应用程序编程环境中，特别是在 PLC 中（见 GB/T20438.6 附录 E），软件架构将由供方作为产品的一种标准特性提供。在这一标准下将要求供方确保用户产品的符合性满足 7.4 的要求。用户使用标准编程工具来定制 PLC 的应用，例如梯形图。7.4.3 至 7.4.8 的要求仍适用。软件架构的定义和文档的要求可作为用户选择适用的 PLC（或类似产品）的信息。

注4：从安全角度讲，软件架构阶段是软件开发中确定基本安全策略的阶段。

注5：尽管 GB/T 20438 标准为由 E/E/PE 安全相关系统实现的安全功能设置了数值化的目标失效量，但系统性安全完整性通常是不可量化的（见 GB/T 20438.4 的 3.5.6），软件安全完整性（见 GB/T 20438.4 的 3.5.5）被定义为置信度为 1-4（见 GB/T 20438.4 的 3.5.9）之间的系统性能力。根据软件的具体使用，本标准将软件失效分为安全的或非安全的。系统/软件架构需要在组件的非安全失效时受到某些架构约束的限制，并且开发方法应考虑这些约束。本标准应用开发和严格的确认技术，使其与要求的系统性能力保持定性的一致。

注6：在选择适当的技术和措施（见附录 A 和 B）来实现本条款的要求时，应考虑软件架构设计的下列属性（见附录 C 属性解释指南，和 GB/T20438.7 的附录 F 非正式定义）：

——有关软件安全要求规范的完整性；

——有关软件安全要求规范的正确性；

——防止固有设计错误 ；

——简单性和易懂性；

——行为的可预见性 ；

——可验证和可测试的设计；

——故障裕度；

——抵御外部事件造成的共因失效。

7.4.3.1 根据软件开发的性质，符合 7.4.4 要求的责任可由多方承担。责任的划分应在安全计划编制中归档（见 GB/T20438.1 的第 6 章）。

7.4.3.2 软件架构设计应由软件供方和/或开发人员来建立并细化。软件架构设计应：

a) 在软件安全生命周期各阶段，为满足软件安全要求规范规定的安全完整性等级应选择和论证一组必要的技术和措施。这些技术和措施包括故障裕度（与硬件一致）和故障避免的软件设计策略，包括（适用时）冗余和多样性；

b) 根据组件/子系统的划分，对每一部分应提供以下信息：

1) 组件/子系统是否已被验证。如果是，它们的验证条件；

2) 每一个子系统/组件是否安全相关；

3) 子系统/组件的软件系统性能力。

c) 确定所有软件/硬件相互作用，并评价和细化它们的重要性；

注：当软件/硬件的相互作用已由系统架构决定时，引用系统架构已经足够。

d) 架构应依据明确定义的或受限于明确定义了特征的符号表示法

e) 选择用于保持所有数据安全完整性的设计特性。数据可包括装置输入/输出数据、通信数据、操作界面数据、维护数据和内部数据库数据；

f) 规定适当的软件架构集成测试，以保证软件架构在要求的安全完整性等级下满足软件安全要求规范。

7.4.3.3 应用 7.4.3.2 后，任何对 E/E/PE 系统安全要求规范的变更请求都应经 E/E/PE 开发人员同意并归档。

注：软件和硬件架构不可避免会有重叠（见图5），因此需要与硬件开发人员讨论可编程电子硬件和软件集成（见7.5）的测试规范等类问题。

7.4.4 支持工具的要求，包括编程语言

注：在选择合适的技术和措施（见附录A和B）来实现本条款的要求时，应考虑支持工具的下列属性（见附录C属性解释指南和GB/T20438.7的附录F非正式定义）：

——工具对生成具有要求属性的软件的支持程度；

——工具的操作和功能的清晰度；

——输出的正确性和可重复性。

7.4.4.1 软件在线支持工具应作为安全相关系统的软件组件来考虑。

注：在线和离线工具的例子见GB/T20438.4的3.2.10和3.2.11。

7.4.4.2 应选择软件离线支持工具作为软件开发活动的密切相关部分。

注1：软件开发生命周期要求见 7.1.2。

注2：在开发过程中，应当使用合适的离线工具支持软件开发，以通过降低引入故障或无法检测到故障的可能性，来增加软件的完整性。软件开发生命周期阶段相关工具的例子包括：

a) 转换或翻译工具，将软件或设计表述（例如文本或图解）从抽象级别转换到另一个级别：设计优化工具、编译器、汇编器、链接器、打包器、装载器和代码生成工具；

b) 验证和确认工具，例如静态代码分析器、测试覆盖监视器、原理证明辅助工具和仿真器等；

c) 诊断工具， 用于在运行条件下维护和监视软件；

d) 基础设施工具，例如开发支持系统；

e) 配置控制工具，例如版本控制工具；

f) 应用数据工具，产生或维护用来定义参数和实例化系统功能的数据。此数据包括功能参数、仪表量程、报警和脱扣限值、失效时应采用的输出状态、布局等。

注3：应选择可集成的离线支持工具，在这种情况下，如果工具能够相互协同工作，则它们可被集成，即一个工具的输出具有合适的内容和格式，用作后一个工具的自动输入，使得在重写中间结果中的人为出错的可能性最小化。

注4：离线支持工具的选择应与应用、安全相关系统和集成工具的需求相匹配。

注5：应考虑在 E/E/PE 安全相关系统的整个生命周期内提供必要服务的适当工具的可用性（如支持规范、设计、实施、建立文档、修改等工具）。

注6：应考虑所选择工具的用户能力。能力要求见 GB/T20438.1 的第 6 章。

7.4.4.3　离线支持工具的选择应得到论证。

7.4.4.4　所有类别为 T2 和 T3 中的离线支持工具应具有规格书或产品文档，用以清晰定义工具的行为和所有使用该工具的指南和约束。见 7.1.2 软件开发生命周期要求和 GB/T 20438.4 的 3.2.11 软件离线支持工具的分类。

注：这个“规格书或者产品文档”并非工具自身符合项安全手册（见GB/T20438.2和本部分的附录D）。符合项安全手册与并入到可执行的安全相关系统的已有组件相关。当通过T3工具生成已有组件，然后并入可执行安全相关系统内，从而所有来自工具的“规格书或者产品文档”的相关信息（例如，优化编译器的文档可能指示不能保证函数参数的求值次序）应该包含在符合项的安全手册中，使得完全或部分依赖并入组件的特定安全功能的完整性评估成为可能。

7.4.4.5　应评估 T2 和 T3 类中的离线支持工具，以确定工具的置信水平和可能影响可执行软件的工具的潜在失效机制。当识别到此失效机制时，应采取合适的缓解措施。

注1：软件 HAZOP 是一种分析潜在软件工具失效后果的技术。

注2：缓解措施的例子包括：避免已知的缺陷、限制工具功能的使用、检查工具输出、为相同目的使用不同工具等。

7.4.4.6　对于 T3 类的每一个工具，应有证据表明工具符合其规范或文档。证据可能基于在类似环境和类似应用（在组织内或其它组织）的成功使用历史和 7.4.4.7 中规定的工具确认的适当组合。

注1：版本历史可提供工具成熟的保证，以及当工具在新开发的环境中使用时，应考虑的错误/歧义的记录。

注2：对 T3 所列的证据也可用于判断 T2 工具结果的正确性。

7.4.4.7　工具确认结果应归档并包括以下内容：

a)　确认活动按时间先后顺序的记录；
b)　所用工具产品手册的版本；
c)　被确认的工具功能；
d)　使用的工具和设备；
e)　确认活动的结果；已归档的确认结果应说明软件已通过确认或说明失败的原因；
f)　测试用例及其后续分析的结果；
g)　预期结果和实际结果的差异。

7.4.4.8　当 7.4.4.6 的一致性证据难以获得时，由工具的错误所导致的可执行安全相关系统的失效，应由有效措施来控制。

注：该措施的一个示例是生成多种冗余代码，允许检测和控制可执行安全相关系统的失效，该失效是由翻译器引入到可执行安全相关系统的错误导致的。

7.4.4.9　应验证集成的成套工具中各种工具之间的兼容性。

注：如果工具能够相互协同工作，则它们可被集成，即一个工具的输出具有合适的内容和格式，用作后一个工具的自动输入，使得在重写中间结果中的人为出错的可能性最小化。见GB/T20438.7中 B3.5。

7.4.4.10　在安全完整性等级要求的范围内，所选的软件或设计描述（包括编程语言）应：

a)　具有一个翻译器，并对翻译器已进行了目的适用性评估，包括在适当时，根据国际和国家标准进行评估；
b)　仅使用已定义的语言特征；
c)　与应用特性匹配；
d)　包含便于检测设计或编程中错误的特征；
e)　支持与设计方法相匹配的特征。

注1：编程语言是一类软件或者设计表述。翻译器将软件或设计表述（例如文本或图解）从一个抽象级别转换到另一级别。翻译器的例子包括：设计优化工具、编译器、汇编器、链接器、打包器、装载器和代码生成工具。

注2：可以为特定的应用项目或一类应用进行翻译器的评估。在后一种情况下，应该给该工具的用户提供预期和适当使用工具的所有必要信息（“规范或产品手册”，见 7.4.4.4）。对特定项目的工具评估可能会简化为检查此项目的工具整体适用性和“规范或产品手册”的符合性（即工具的正确使用）。正确使用可能包括在此特定项目中额外的验证活动。

注3：确认套件（即正确翻译的一组预知的测试程序）可用于根据定义的准则评估翻译器的适用性，其中包括功能和非功能的要求。对于功能的翻译器要求，动态测试可能是主要的确认技术。如果可能，应使用自动测试套件。

7.4.4.11 当 7.4.4.10 不能被完全满足时，应论证语言目的的适合性和所有处理已识别的语言缺陷的额外措施。

7.4.4.12 开发所有安全相关软件的编程语言应根据合适的编程语言编码标准来使用。

注：关于软件安全编码的标准指南,见GB/T20438.7。

7.4.4.13 编程语言编码标准须规定良好的编程习惯，禁止不安全的语言特性（例如，未定义的语言特性，非结构化设计等），促进代码的可读性，便于验证和测试，并规定源代码建立文档的规程。当可行时，源代码中应包含下列信息：

a) 法人实体（例如公司，作者等）；
b) 描述；
c) 输入和输出；
d) 配置管理历史。

7.4.4.14 使用代码自动生成或类似的自动翻译，应在开发生命周期选择开发支持工具这一节点时，评估安全相关系统开发所采用的自动翻译程序的适用性。

7.4.4.15 当 T2 和 T3 类的离线支持工具配置基线中产生项时，配置管理应确保工具上的信息被记录在配置基线中。尤其应包括：

a) 工具和其版本的标识；
b) 对应使用工具版本的配置基线项的标识；
c) 每个配置基线项的工具使用方式（包括选定的工具参数、选项和脚本）。

注：本条的目的是允许基线被重构。

7.4.4.16 配置管理应确保仅使用 T2 和 T3 类中的工具合格的版本。

7.4.4.17 配置管理应确保仅使用相互兼容及与安全相关系统兼容的工具。

注：安全相关系统硬件也可能对软件工具施加兼容性限制，例如，处理器的仿真器需要是一个真实处理器电路的精确模型。

7.4.4.18 每一个新版本的离线支持工具应确认是否适用。如果提供了以下充分证据，这个适用性可以早期版本提供的证据为证。

a) 功能上的差异（若存在的话）将不会影响与工具集中其它工具的兼容性；
b) 新版本不大可能包含重大的新的未知错误。

注：新版本不大可能包含重要的新的未知错误的证据，可基于(1)所作更改的清晰识别，(2)对新版本的验证和确认行为的分析，(3)以及任何来自其它用户与新版本有关的实际操作经验。

7.4.4.19 依据软件开发的性质，对 7.4.4 的符合性可由多方负责。责任的划分应在安全计划编制期间（见 GB/T20438.1 的第 6 章）形成文档。

7.4.5 详细设计和开发要求-软件系统设计

注1：此处定义的详细设计指软件系统设计：架构中的主要组件划分为一个软件模块系统；各自的软件模块设计；编码。在小型应用中，软件系统设计和架构设计可整合在一起。

注2：详细设计和开发的性质因软件开发活动和软件架构性质的不同而变化（见 7.4.3）。在某些应用程序编程环境中，例如梯形图和功能块，详细设计可看作是配置而不是编程。然而，以结构化方法设计软件仍是一种良好的实践，包括将软件组织成模块化的结构，并（尽可能地）分离出安全相关部分；包括范围检测和其它预防数据输入错误的特性；使用以往确认过的软件模块；和提供便于未来软件修改的设计。

注3：为选择合适的技术和措施（见附录 A 和 B）以实现本章的要求，以下属性（见附录 C 的属性解释指南，和 GB/T 20438.7 附录 F 中的非正式定义）在设计与开发中应予考虑：

——有关软件安全要求规范的完整性；

——有关软件安全要求规范的正确性；

——防止固有设计故障；

——简单性和易懂性；

——行为可预见性；

——可验证和可测试的设计；

——故障裕度/故障检测；

——防止共因失效。

7.4.5.1 根据软件开发的性质，7.4.5 的符合性责任可由多方承担。责任的划分应在安全计划编制中归档（见 GB/T20438.1 第 6 章）。

7.4.5.2 下列信息应在详细设计开始前获得：E/E/PE 安全相关系统的要求规范；软件架构设计；系统安全相关软件方面的确认计划。

7.4.5.3 所生成的软件应具有：模块化、可测试性、可安全修改的能力。

7.4.5.4 对于软件架构设计中的每一个主要组件/子系统，设计的进一步细化应基于软件模块的划分（即软件系统设计的规范）。应规定每个软件模块的设计以及对每个软件模块的验证。

注1：对于已有软件组件，见 7.4.2。

注2：验证包含测试和分析。

7.4.5.5 应规定适当的软件系统集成测试，以保证软件系统满足在所要求安全完整性等级上的软件安全要求规范。

7.4.6 代码实现要求

注：为达到安全完整性等级的要求，源代码应具有以下代码应考虑的属性（见附录A和B的技术规范，和附录C的属性解释指南）：

a) 可读、可理解和可测试；

b) 满足软件模块设计的规定要求（见 7.4.5）；

c) 满足编码标准的规定要求（见 7.4.4）；

d) 满足安全计划编制中规定的所有相关要求（见第 6 章）。

7.4.6.1 每一软件代码模块应复审。当通过自动化工具产生代码时，应符合 7.4.4 的要求。源代码包含了复用的已有软件时，应满足 7.4.2 的要求。

注：代码复审是一种验证活动（见7.9）。代码复审可依靠多种代码检查方式来完成：(1)由个人；（2）通过软件走查（见GB/T20438.7的C.5.15）；(3)通过形式化检查（见GB/T20438.7的C.5.14）。以上方法严格程度递增。

7.4.7 软件模块测试要求

注1：测试软件模块正确地满足了测试规范是一种验证活动（见 7.9），是代码复审和软件模块测试的结合，用以确保软件模块满足其相关规范，即已验证。

注2：为选择合适的技术与措施（见附录 A 和 B）以实现本章的要求，应考虑软件模块测试的下列属性（属性解释指南见附录 C 和 GB/T20438.7 附录 F 中非正式定义）：

——针对软件设计规范，所做测试的完整性；

——针对软件设计规范（成功完成），所做测试的正确性；

——可重复性；

——精确定义的测试配置。

7.4.7.1 每一软件模块均应根据软件模块测试规范的要求进行验证，该测试规范是在软件系统设计阶段制定的（见 7.4.5）。

注：验证包括测试和分析。

7.4.7.2 这个验证应表明每一软件模块是否执行预定功能且不执行非预定功能。

注1：这不意味着测试所有输入组合和输出组合。测试所有的等价类或结构可能就已经足够了，边界值分析或控制流分析可将测试用例减少至一个可接受的数量。可分析的程序能更容易地满足要求。这些技术见 GB/T20438.7 的附录 C。

注2：当开发使用了形式化方法，形式化证明或断言时，此类测试可在一定范围内减少。这些技术见 GB/T20438.7 的附录 C。

注3：尽管系统性安全完整性通常无法量化（见 GB/T20438.4 的 3.5.6），如果对统计学有效证据的所有相关条件均得到满足，量化的统计证据是可接受的，例如，见 GB/T20438.7 中的附录 D。

注4：如果模块足够简单，使得进行详尽测试是可行的，那么这可能是证明符合性的最有效方式。

7.4.7.3 软件模块测试的结果应归档。

7.4.7.4 应规定未通过测试的校正措施规程。

7.4.8 软件集成测试的要求

注：对软件的是否正确集成进行测试是一种验证活动（见7.9）。

7.4.8.1 软件集成测试应在设计和开发阶段予以规定（见 7.4.5）。

7.4.8.2 软件系统集成测试规范应规定以下内容：

a) 将软件划分为可管理的集成集；

b) 测试用例和测试数据；

c) 执行测试的类型；

d) 测试环境、工具、配置和程序；

e) 判定测试完成的准则；

f) 测试失败的校正措施规程。

7.4.8.3 软件应根据软件系统集成测试规范中规定的软件集成测试要求进行测试。这些测试应表明所有软件模块和软件组件/子系统正确交互以执行其预定的功能而不执行非预定的功能。

注1：这不意味着测试所有输入组合和输出组合。测试所有的等价类或结构可能就已经足够了，边界值分析或控制流分析可将测试用例减少至一个可接受的数量。可分析的程序能更容易地满足要求。这些技术见 GB/T20438.7 的附录 C。

注2：当开发使用形式化方法、形式化证明或断言时，此类测试可在一定范围内减少。有关这些技术见 GB/T20438.7 的附录 C。

注3：尽管系统性安全完整性通常无法量化（见 GB/T20438.4 的 3.5.6），如果对统计学有效证据的所有相关条件均得到满足，量化的统计证据是可接受的，例如，见 GB/T20438.7 中的附录 D。

7.4.8.4 软件集成测试的结果应归档、说明测试结果及是否满足目的和测试准则。若存在失败的集成测试，应记录失败的原因。

7.4.8.5 在软件集成过程中，应对软件的任何修改进行影响分析以确定所有受影响的软件模块和所必需的再验证和再设计活动。

7.5 可编程电子集成（硬件和软件）

注：这一阶段是图4中的方框10.4。

7.5.1 目的

7.5.1.1 本条的第一个目的是在目标可编程电子硬件上集成软件。

7.5.1.2 本条的第二个目的是将软件和硬件结合到安全相关可编程电子上，以保证其兼容性和满足预定安全完整性等级的要求。

注1：测试软件是否正确集成到可编程电子硬件中是一种验证活动（见 7.9）。

注2：根据应用的性质这些活动可与 7.4.8 结合。

7.5.2 要求

注：为选择合适的技术和措施（见附录A和B）来实现本章的要求，应考虑集成的下列属性（见附录C的属性解释指南，和GB/T 20438.7附录F中的非正式定义）：

——针对设计规范，所做集成的完整性；

——针对设计规范（成功完成），所做集成的正确性；

——可重复性；

——精确定义的集成配置。

7.5.2.1 应在设计和开发阶段（见 7.4.3）中规定集成测试，以确保在安全相关可编程电子中硬件和软件的兼容性。

注：可能会要求与E/E/PE系统开发人员紧密合作以开展集成测试。

7.5.2.2 软件/PE 集成测试规范（硬件和软件）应规定以下内容：

a) 将系统拆分为各集成级；
b) 测试用例和测试数据；
c) 执行测试的类型；
d) 测试环境包括工具、支持软件和配置描述；
e) 判定测试完成的准则。

7.5.2.3 软件/PE 集成测试规范（硬件和软件）应区分：可由开发人员在其场地完成的活动和需要访问用户现场的活动。

7.5.2.4 软件/PE 集成测试规范（硬件和软件）应区分以下活动：

a) 将软件系统纳入目标可编程电子硬件；
b) E/E/PE 集成，即加入接口如传感器和执行器；
c) 将 E/E/PE 安全相关系统用到 EUC 中。

注：b)和c)已由GB/T20438.1和GB/T20438.2覆盖，此处为保证完整性将a）包括进来。这些一般不是软件开发人员的职责。

7.5.2.5 应根据软件/PE 集成测试规范（硬件和软件）对软件和安全相关可编程电子硬件进行集成。

7.5.2.6 在安全相关可编程电子（硬件和软件）集成测试中，应对集成系统的所有变更进行影响分析。影响分析应确定所有受影响的软件模块和必要的再验证活动。

7.5.2.7 为便于后续分析，测试用例及其预期结果应归档。

7.5.2.8 安全相关可编程电子（硬件和软件）的集成测试应归档，说明测试结果以及是否满足测试目的和测试准则。如果出现失败，应将失败原因归档。软件的所有修改或变更都应进行影响分析，以确定对所有受影响的软件组件/模块，以及必要的再验证和再设计活动。

7.6 软件操作和修改规程

注：该阶段是图4中方框10.5。

7.6.1 目的

本条要求的目的是提供软件有关必要的信息和规程以保证在操作和修改阶段中保持E/E/PE安全相关系统的功能安全。

7.6.2 要求

要求在本部分的7.8和GB/T20438.2的7.6中给出。

注：在本标准中，软件（不同于硬件）不能进行维护，而总是进行修改。

7.7 系统安全确认的软件方面

注1：这一阶段为图 4 中方框 10.6。

注2：软件确认通常不能脱离它相关的硬件和系统环境。

7.7.1 目的

本条的目的是保证集成的系统在所需的安全完整性等级上符合软件安全要求规范。

7.7.2 要求

注：为选择合适的技术和措施（见附录A和B）来实现本章的要求，应考虑安全确认的下列属性（见附录C属性解释指导和GB/T20438.7的附录F非正式定义）：

——针对软件设计规范，所做确认的完整性；

——针对软件设计规范（成功完成），所做确认的正确性；

——可重复性；

——精确定义的确认配置。

7.7.2.1 如果在 E/E/PE 安全相关系统的安全确认计划中已建立了对安全相关软件的符合性要求（见 GB/T20438.2 中 7.7），则不必重复确认。

7.7.2.2 确认活动应根据系统安全的软件方面安全确认计划的规定进行。

7.7.2.3 根据软件开发的性质，7.7 的符合性要求的责任可由多方承担。责任的划分应在安全计划编制中归档（见 GB/T20438.1 第 6 章）。

7.7.2.4 系统安全的软件方面确认的结果应归档。

7.7.2.5 对每一个安全功能，软件安全确认应对以下结果文档化：

a) 确认活动的按时间顺序的记录，以便追溯活动的顺序；

注：记录测试结果时，重要的是能够追溯活动的顺序。此要求的重点是追溯一个活动序列，而不是仅产生一个时间/日期的文档列表。

b) 所用的系统安全的软件方面确认计划（见 7.3）的版本；

c) 根据系统安全的软件方面确认计划，被确认（通过测试或分析）的安全功能；

d) 使用的工具和设备及其校准数据；

e) 确认活动的结果；

f) 实际结果和预期结果的差异。

7.7.2.6 当实际结果和预期结果出现差异时，所进行的分析和对是继续确认或是提出变更请求并返回开发生命周期较早阶段所做的决定，都应作为系统安全的软件方面的确认结果文档化。

注：7.7.2.2至7.7.2.6的要求基于GB/T20438.1中7.14的一般要求。

7.7.2.7 系统安全的安全相关软件方面的确认应符合下列要求：

a) 测试应该是软件确认的主要方法；分析、动画和建模可用作确认活动的补充；

b) 软件应通过对以下情形的仿真进行测试：

——正常操作中出现的输入信号；

——预期的事件；

——不期望条件下要求的系统动作。

c) 供方和/或开发人员（或对符合性负责的各方）应使系统开发者得到系统安全的软件方面确认的文档化结果及其所有的附属文档，以使其产品满足 GB/T20438.1 和 GB/T20438.2 的要求。

7.7.2.8 软件工具应符合 7.4.4 的要求。

7.7.2.9 系统安全的安全相关软件方面的确认结果应满足以下要求：

a) 测试应表明所有安全相关软件规定的要求都正确满足，并且软件不执行非预定的功能；

b) 测试用例及其结果应文档化，以用于后续的分析和由安全完整性等级所要求的独立评估（见 GB/T20438.1 的第 8 章）；

c) 文档化的系统安全的软件方面确认的结果应表明(1)软件已通过确认，或(2)未通过确认的原因。

7.8 软件修改

注：这一阶段是图4中的方框10.5。

7.8.1 目的

本条的目的是指导修正、增强或调整已确认软件，以保证维持要求的软件系统性能力。

7.8.2 要求

注：为选择合适的技术和措施（见附录A和B）来实现本章的要求，应考虑软件修改的下列属性（见附录C属性解释指导和GB/T20438.7的附录F非正式定义）：

——针对要求，所做修改的完整性；

——针对要求，所做修改的正确性；

——防止引入固有设计故障；

——避免不需要的行为；

——可验证和可测试的设计；

——回归测试和验证覆盖。

7.8.2.1 在执行任何软件修改之前，软件修改规程应已可用（见 GB/T20438.1 中的 7.16）。

注1：7.8.2.1 至 7.8.2.9 主要用于在软件运行阶段中发生的改变。它们也可用于可编程电子集成和整体安装和调试阶段（见 GB/T20438.1 中 7.13）。

注2：一个修改规程模型的例子见 GB/T20438.1 中的图 9。

7.8.2.2 只有经批准的软件修改请求按照安全计划编制阶段规定的规程（见第 6 章）发出时，才应启动修改，修改请求包括以下内容：

a) 可能受到影响的危险；

b) 建议的修改；

c) 修改的原因。

注：修改请求原因的例子如下：

——发现功能安全低于安全要求规范的要求；

——系统性故障经验；

——新的或修订的安全法规；

——EUC 或其使用的修改；

——整体安全要求的修改；

——操作和维护性能分析，表明性能低于目标值；

——例行功能安全审核。

7.8.2.3 对 E/E/PE 安全相关系统的功能安全所建议的软件修改，应进行影响分析，以：

a) 确定是否需要危险和风险分析；

b) 确定哪个软件安全生命周期阶段需重复。

7.8.2.4 在 7.8.2.3 中所做影响分析的结果应文档化。

7.8.2.5 所有对 E/E/PE 安全相关系统的功能安全有影响的修改应返回软件安全生命周期的一个相应阶段。所有后续的阶段也应根据本标准对特定阶段要求规定的规程执行。安全计划编制（见第 6 章）应细化所有后续的活动。

注：可能需要执行一次全面的危险和风险分析，此分析可能产生与当前规定的、由E/E/PE安全相关系统实现的安全功能不一样的安全完整性等级要求。

7.8.2.6 安全相关软件修改的安全计划编制应满足 GB/T20438.1 中第 6 章的要求，特别是：

a) 人员识别和其所需能力的规范；

b) 修改的详细规范；

c) 验证计划编制；

d) 根据安全完整性等级要求的程度，进行再确认的范围和修改的测试。

注：根据应用的性质，领域内专家的参与可能很重要。

7.8.2.7 修改应按照计划执行。

7.8.2.8 所有修改的细节应被文档化，包括参考：

a) 修改/改型请求；

b) 评估建议的软件修改对功能安全影响的分析结果和决策及其相关的论证；

c) 软件配置管理历史；

d) 与正常操作和条件的偏差；

e) 受修改活动影响的所有文档化信息。

7.8.2.9 所有修改的详细信息应文档化。文档应包括数据和结果的再验证和再确认。

7.8.2.10 对所需修改或改型活动的评估应取决于影响分析的结果和软件系统性能力。

7.9 软件验证

7.9.1 目的

本条的目的是，针对安全完整性等级要求的程度，测试和评估软件安全生命周期在给定阶段的输出，保证该阶段输出对于相应输入的正确性和一致性。

注1：本条考虑安全生命周期几个阶段通用的验证内容。本条不对在 7.4.7（软件模块测试）、7.4.8(软件集成)和 7.5（可编程电子集成）中的验证测试组件提出额外要求，因为这些本身就是验证活动。本条也不要求对软件确认（见 7.7）的额外验证，因为本标准中软件确认是符合安全要求规范的证明。对安全要求规范自身是否正确的检查由领域内专家完成。

注2：根据软件架构，验证活动的责任可在所有参与软件开发和修改的组织间进行划分。

7.9.2 要求

注：在选择合适的技术和措施（见附录A和B）来实现本章的要求时，应考虑数据验证的下列属性（见附录C属性解释指导和GB/T20438.7的附录F非正式定义）：

——针对前一阶段，所做验证的完整性；

——针对前一阶段（成功完成），所做验证的正确性；

——可重复性；

——精确定义的验证配置。

7.9.2.1 对软件安全生命周期的每一个阶段，应与开发过程同步制定软件验证计划（见 7.3），并且对软件验证文档化。

7.9.2.2 软件验证计划编制应涉及验证活动中使用的准则、技术和工具，并应指明：

a) 安全完整性要求的评价；

b) 验证策略、活动和技术的选择及其文档化；

c) 验证工具（测试工具、专用的测试软件、输入/输出仿真器等）的选择和使用；

d) 验证结果的评价；

e) 采用的纠正措施 。

7.9.2.3 软件验证应根据计划执行。

注：验证技术和措施的选择以及验证活动的独立程度取决于很多因素并可能在应用领域的标准中规定。因素的例子包括：

——项目规模；

——复杂程度；

——设计的新颖程度；

——技术的新颖程度。

7.9.2.4 应对证据文档化，以表明对阶段的验证从各个方面都已圆满完成。

7.9.2.5 每次验证后，验证文档应包括：

a) 被验证项的识别；

b) 完成验证所依据信息的识别；

注1：开展验证所依据的信息包括但不限于：来自前一生命周期的输入、设计标准、编码标准和所用工具。

c) 不符合项。

注2：不符合项的例子包括软件模块、数据结构和不适于问题的算法。

7.9.2.6 软件安全生命周期阶段 N 中为阶段 N+1 正确执行所需的所有基本信息都应可获得并被验证，阶段 N 的输出包括：

a) 针对以下内容，阶段 N 的规范、设计或代码的充分性：

1) 功能性；

2) 安全完整性、安全计划编制的执行和其它要求（见第 6 章）；

3) 对于开发团队而言的可读性；

4) 进一步验证的可测试性；

5) 安全修改，以允许进一步改进。

b) 针对阶段 N 设计的规定和描述，确认计划编制和/或规定测试的充分性。

c) 针对下列内容之间不一致性的检查：

1) 在阶段 N 中规定的测试和在前一阶段 N-1 中规定的测试；

2) 阶段 N 中的各个输出。

7.9.2.7 根据选择的软件开发生命周期（见 7.1），应执行下列验证活动：

a) 软件安全要求的验证；
b) 软件架构验证；
c) 软件系统设计验证；
d) 软件模块设计验证；
e) 代码验证；
f) 数据验证；
g) 时间性能的验证；
h) 软件模块测试（见 7.4.7）；
i) 软件集成测试（见 7.4.8）；
j) 可编程电子集成测试（见 7.5）；
k) 系统安全的软件方面确认（见 7.7）。

注：对于a)到g)的要求见下文。

7.9.2.8 软件安全要求验证：在完成软件安全要求规范后，并且在接下来的软件设计和开发阶段开始前，验证应：

a) 考虑软件安全要求规范是否已充分满足 E/E/PE 系统安全要求规范（见 GB/T20438.1 的 7.10 和 GB/T20438.2 的 7.2）对功能性、安全完整性、安全计划编制的执行和任何其它要求；
b) 考虑系统安全的软件方面的确认计划是否充分满足软件安全要求规范；
c) 检查下列内容之间的不一致性:
   1) 软件安全要求规范和 E/E/PE 系统安全要求规范（见 GB/T20438.1 的 7.10 和 GB/T20438.2 的 7.2）；
   2) 软件安全要求规范和系统安全的软件方面确认计划。

7.9.2.9 软件架构验证：在完成软件架构设计后，验证应：

a) 考虑软件架构设计是否充分满足软件安全要求规范；
b) 考虑软件架构设计规定的集成测试是否充分；
c) 针对以下方面，考虑每一主要组件/子系统的属性是否充分：
   1) 所需的安全性能的可行性；
   2) 进一步验证的可测试性；
   3) 对于开发团队而言的可读性；
   4) 安全修改，以允许进一步改进。
d) 检查以下内容之间的不一致性：
   1) 软件架构设计和软件安全要求规范；
   2) 软件架构设计及其集成测试；
   3) 软件架构设计集成测试和系统安全的软件方面确认计划。

7.9.2.10 软件系统设计验证：在完成软件系统设计后，验证应：

a) 考虑软件系统设计（见 7.4.5）是否充分满足软件架构设计；
b) 考虑软件系统集成规定的测试（见 7.4.5）是否已充分满足软件系统设计（见 7.4.5）；
c) 针对以下方面，考虑软件系统设计规范的每一主要组件的属性（见 7.4.5）是否足够：
   1) 要求的安全性能的可行性；
   2) 进一步验证的可测试性；
   3) 对于开发团队而言的可读性；
   4) 安全修改，以允许进一步改进。

注：软件系统集成测试可规定为软件架构集成测试的一部分。

d) 检查以下内容之间的不一致性：
   1) 软件系统设计规范（见 7.4.5）和软件架构设计；
   2) 软件系统设计规范（见 7.4.5）和软件系统集成测试规范（见 7.4.5）；
   3) 软件系统集成测试规范规定的测试（见 7.4.5）和软件架构集成测试规范的测试（见 7.4.3）。

7.9.2.11 软件模块设计验证：在完成每一软件模块设计后，验证应：
a) 考虑软件模块设计规范（见 7.4.5）是否已充分满足软件系统设计规范（见 7.4.5）；
b) 考虑软件模块测试规范（见 7.4.5）是否已充分满足软件模块设计规范（见 7.4.5）；
c) 针对以下方面，考虑每一软件模块的属性是否足够：
   1) 要求的安全性能的可行性（见软件安全要求规范）；
   2) 进一步验证的可测试性；
   3) 对于开发团队而言的可读性；
   4) 安全修改，以允许进一步改进。
d) 检查以下内容之间的不一致性：
   1) 软件模块设计规范（见 7.4.5）和软件系统设计规范（见 7.4.5）；
   2) （对每一个软件模块）软件模块设计规范（见 7.4.5）和软件模块测试规范（见 7.4.5）；
   3) 软件模块测试规范（见 7.4.5）和软件系统集成测试规范（见 7.4.5）。

7.9.2.12 代码验证：源代码需通过静态方法验证，以确保符合软件模块设计规范（见 7.4.5）、要求的编码标准（见 7.4.4）和系统安全的软件方面确认计划。

注：在软件安全生命周期的早期阶段，验证是静态的（如检查、复审、形式化证明等）。代码验证包括软件检查和走查等技术。代码验证和软件模块测试结论的结合保证每一软件模块满足其相关规范。此后，测试成为验证的主要方法。

7.9.2.13 数据验证。
a) 应验证数据结构。
b) 应验证应用数据：
   1) 与数据结构的一致性；
   2) 针对应用要求的完整性；
   3) 与相关的系统软件的兼容性（如执行的序列、运行时等）；
   4) 数据值的正确性。
c) 应针对应用要求验证所有运行参数。
d) 应针对以下内容，验证所有设备接口及其相关软件（即传感器、执行器和离线接口，见 7.2.2.12）：
   1) 预期接口失效的检测；
   2) 预期接口失效的容错。
e) 所有通信接口及其相应软件应以一个充分的等级进行验证：
   1) 失效检测；
   2) 防范损坏；
   3) 数据确认。

7.9.2.14 时间性能的验证：应验证在时间域的行为可预测性。

注：时间行为可能包括：性能、资源、响应时间、最坏情况下的执行时间、超负荷、无死锁、运行时系统等。

8 功能安全评估

注：在选择合适的技术和措施（见附录A和B）来实现本章的要求时，应考虑功能安全评估的下列属性（见附录C属性解释指导和GB/T20438.7的附录F非正式定义）：

——针对本标准，功能安全评估的完整性；

——针对设计规范（成功完成），功能安全评估的正确性；

——对所有发现问题的可追溯的闭环；

——不需要大量的重新评估的变更后，修改功能安全评估的能力；

——可重复性；

——及时性；

——精确定义的配置。

8.1 GB/T20438.1 第 8 章中的目的和要求适用于安全相关软件的评估。

8.2 除非在应用领域国家标准中另有规定，执行功能安全评估的有关方面的最低独立性水平应与 GB/T20438.1 中第 8 章的规定一致。

8.3 功能安全评估可利用表 A10 中活动的结果。

注：选择附录A和附录B中的技术其自身并不能保证达到要求的安全完整性（见7.1.2.7），评估方还需考虑：

——在整个开发周期中选择的方法、语言和工具的一致性和互补性；

——开发人员是否选择了他们完全理解的方法、语言和工具；

——选择的方法、语言和工具在开发阶段中对所针对的特定问题是否普遍适用。

附 录 A
（规范性附录）
技术和措施选择指导

本部分中一些条款有相关表格，如7.2（软件安全要求规范）与表A.1有关，附录B中更详细的表有些是由附录A中的表项扩展来的，如表B.2根据表A.5中动态分析和测试的主题扩展而成。

附录A和附录B中提及的技术和措施可见GB/T20438.7。

表中的每一技术和措施都有安全完整性等级1到4的推荐，推荐如下：

HR	在该安全完整性等级下极力推荐的技术或措施。如未采用这种技术或措施则应在安全计划中详细记录未使用该技术和措施的理由，并需经评估方认可。
R	在该安全完整性等级下低于 HR 推荐程度的所推荐的技术或措施。
---	不推荐或不反对使用的技术或措施。
NR	在该安全完整性等级下明确不推荐的技术或措施。如果采用这类技术或措施，应在安全计划中参照附录 C 详细记录使用该技术和措施的理由，并需经评估方认可。

应根据安全完整性等级选择适当的技术/措施。可选择的或等价的技术/措施用数字后跟字母的方式表示。只需满足一种可选择的或等价的技术/措施。

假如已经满足要求及目标，可应用其他措施和技术。选择技术指导参见附录C。

技术和措施的分级与GB/T20438.2中所使用的有效性的概念有关。在所有其他因素都相同的情况下，HR类的技术在软件开发中防止引入系统性故障或在软件执行中控制残余故障方面（针对软件架构）都比R类技术更加有效。

考虑到大量影响软件系统性能力的因素，不可能给出一种将技术和措施结合起来的、对任何指定的应用都正确的算法。附录C给出了选择具体技术来达到软件系统性能力要求的原理指导。

对于某个特定的应用，在安全计划编制中应阐明技术或措施的适当结合。除非在表中的注做出了其它要求，否则应选择相应的技术或措施。

GB/T20438.6中以两个实例的形式对表格的解释给出了初步的指导。

表A.1 软件安全要求规范（见 7.2）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1a	半形式化方法	表 B.7	R	R	HR	HR
1b	形式化方法	B.2.2, C.2.4	---	R	R	HR
2	在系统安全要求和软件安全要求间向前可追溯	C.2.11	R	R	HR	HR
3	在安全要求和认识到的安全需求间向后可追溯	C.2.11	R	R	HR	HR
4	支持上述适当的技术或措施的计算机辅助规范工具	B.2.4	R	R	HR	HR

注1：软件安全要求规范应使用自然语言和反映应用的必要的数学符号对问题进行描述。

注2：本表反映了为清晰和准确说明软件安全要求的附加要求。

注3：见表C.1。

注4：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表A.2　软件设计和开发：软件架构设计（见 7.4.3）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
	架构与设计特点					
1	故障检测	C.3.1	---	R	HR	HR
2	错误检测代码	C.3.2	R	R	R	HR
3a	失效断言编程	C.3.3	R	R	R	HR
3b	多样化监视技术（同一台计算机上的监视功能和被监视功能之间独立）	C.3.4	---	R	R	---
3c	多样化监视技术（监视计算机与被监视计算机之间分离）	C.3.4	---	R	R	HR
3d	多样化冗余，实现相同软件安全要求规范	C.3.5	---	---	---	R
3e	功能上多样化冗余，实现不同软件安全要求规范	C.3.5	---	---	R	HR
3f	后向恢复	C.3.6	R	R	---	NR
3g	无状态软件设计（或者有限状态设计）	C.2.12	---	---	R	HR
4a	故障恢复重试机制	C.3.7	R	R	---	---
4b	适度降级	C.3.8	R	R	HR	HR
5	人工智能——故障纠正	C.3.9	---	NR	NR	NR
6	动态再配置	C.3.10	---	NR	NR	NR
7	模块化方法	表 B.9	HR	HR	HR	HR
8	使用可信赖/已证实的软件组件（如可获得）	C.2.10	R	HR	HR	HR
9	在软件安全要求规范和软件架构间向前的追溯性	C.2.11	R	R	HR	HR
10	在软件安全要求规范和软件架构间向后的追溯性	C.2.11	R	R	HR	HR
11a	结构化的图解方法[b]	C.2.1	HR	HR	HR	HR
11b	半形式化方法[b]	表 B.7	R	R	HR	HR
11c	形式化设计和优化方法[b]	B.2.2, C.2.4	---	R	R	HR
11d	自动软件生成	C.4.6	R	R	R	R
12	计算机辅助规范和设计工具	B.2.4	R	R	HR	HR
13a	周期性运转，并且确定最大周期时间	C.3.11	R	HR	HR	HR
13b	时间触发式架构	C.3.11	R	HR	HR	HR
13c	事件驱动，并且确定最大响应时间	C.3.11	R	HR	HR	-
14	静态资源分配	C.2.6.3	-	R	HR	HR
15	访问共享资源的静态同步	C.2.6.3	-	-	R	HR

注1：表 A.2 中给出的一些方法是关于设计理念的，其余的是关于如何表达设计的。

注2：GB/T 20438.2 表中与故障裕度（失效控制）相关的措施应与架构要求和可编程电子设备的硬件失效控制一起考虑。

注3：见表 C.2。

注4：第 13 组措施仅适用于有安全时间要求的系统及软件。

注5：第 14 组措施使用动态对象（例如使用执行堆栈或堆）可能会对可用内存和执行时间上施加要求。在以下情况下，第 14 组措施不需要应用，即如果使用的编译器能够确保：a）在运行前，为所有的动态变量和对象分配了足够的内存，或者保证一旦发生内存分配错误，能够实现安全状态；b）其响应时间满足要求。

注6：措施 4a。故障恢复重试往往在任何的 SIL 中都是适用的，但应设定一个重试次数的限值。

注7：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了 GB/T20438.7 中附录 B 和附录 C 给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

[b] 第 11 组，“结构化方法”。在 SIL3 或 SIL4 的情况下，仅当 11b 不适合时，才使用措施 11a。

表A.3　软件设计和开发：支持工具和编程语言（见 7.4.4）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	适当的编程语言	C.4.5	HR	HR	HR	HR
2	强类型编程语言	C.4.1	HR	HR	HR	HR
3	语言子集	C.4.2	---	---	HR	HR
4a	已认证的工具和已认证的翻译器	C.4.3	R	HR	HR	HR
4b	工具和翻译器：通过使用提高置信度	C.4.4	HR	HR	HR	HR

注1：见表C.3

注2：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表A.4　软件设计和开发：详细设计（见 7.4.5 和 7.4.6）

（包括软件系统设计、软件模块设计和编码）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1a	结构化方法[b]	C.2.1	HR	HR	HR	HR
1b	半形式化方法[b]	表 B.7	R	HR	HR	HR
1c	形式化设计和优化方法[b]	B.2.2，C.2.4	---	R	R	HR
2	计算机辅助设计工具	B.3.5	R	R	HR	HR
3	防御性编程	C.2.5	---	R	HR	HR
4	模块化方法	表 B.9	HR	HR	HR	HR
5	设计和编码标准	C.2.6，表 B1	R	HR	HR	HR
6	结构化编程	C.2.7	HR	HR	HR	HR
7	使用可信的/经验证的软件组件（如可获得）	C.2.10	R	HR	HR	HR

8	在软件安全要求规范和软件设计间向前可追溯	C.2.11	R	R	HR	HR

注1：见表C.4。

注2：针对安全相关系统的面向对象软件开发的适合性还有争论。面向对象的架构和设计的指导见GB/T 20438.7中附录G。

注3：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

[b] 第 1 组，“结构化方法”。在 SIL3 或 SIL4 的情况下，仅当 11b 不适合时，才使用措施 11a。

表A.5 软件设计和开发：软件模块测试和集成（见 7.4.7 和 7.4.8）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	概率测试	C.5.1	---	R	R	R
2	动态分析和测试	B.6.5 表 B.2	R	HR	HR	HR
3	数据记录和分析	C.5.2	HR	HR	HR	HR
4	功能和黑盒测试	B.5.1 B.5.2 表 B.3	HR	HR	HR	HR
5	性能测试	表 B.6	R	R	HR	HR
6	基于模型的测试	C.5.27	R	R	HR	HR
7	接口测试	C.5.3	R	R	HR	HR
8	测试管理和自动化工具	C.4.7	R	HR	HR	HR
9	在软件设计规范与模块及集成测试规范间向前可追溯	C.2.11	R	R	HR	HR
10	形式化验证	C.5.12	---	---	R	R

注1：软件模块和集成测试是验证活动（见表B.9）。

注2：见表C.5。

注3：技术9中形式化验证可能减少模块及集成测试所要求的数量和程度。

注4：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。

表A.6 可编程电子集成（硬件和软件）（见 7.5）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	功能和黑盒测试	B.5.1 B.5.2 表 B.3	HR	HR	HR	HR
2	性能测试	C.5.20 表 B.6	R	R	HR	HR

3	在系统及针对硬件/软件集成的软件设计要求和硬件/软件集成测试规范之间向前可追溯	C.2.11	R	R	HR	HR
注1：可编程电子集成是一种验证活动（见表A.9）。 注2：见表C.6。 注3：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。						

表A.7 系统安全确认的软件方面（见7.7）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	概率测试	C.5.1	---	R	R	HR
2	过程仿真	C5.18	R	R	HR	HR
3	建模	表B.5	R	R	HR	HR
4	功能和黑盒测试	B.5.1 B.5.2 表B.3	HR	HR	HR	HR
5	在软件安全要求规范和软件安全确认计划间向前可追溯	C2.11	R	R	HR	HR
6	在软件安全确认计划和软件安全要求规范间向后可追溯	C.2.11	R	R	HR	HR
注1：见表C.7。 注2：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。						

表A.8 修改（见7.8）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	影响分析	C.5.23	HR	HR	HR	HR
2	再验证被变更的软件模块	C.5.23	HR	HR	HR	HR
3	再验证受影响的软件模块	C.5.23	R	HR	HR	HR
4a	再确认整个系统	表A.7	---	R	HR	HR
4b	回归确认	C.5.25	R	HR	HR	HR
5	软件配置管理	C.5.24	HR	HR	HR	HR
6	数据记录和分析	C.5.2	HR	HR	HR	HR
7	在软件安全要求规范和软件修改计划（包括再验证和再确认）之间向前可追溯	C.2.11	R	R	HR	HR
8	软件修改计划（包括再验证和再确认）和软件安全要求规范之间向后可追溯	C.2.11	R	R	HR	HR

注1：见表C.8。

注2：第4组技术中影响分析是回归确认的必需部分。见GB/T 20438.7。

注3：第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表A.9　软件验证（见 7.9）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	形式化证明	C.5.12	---	R	R	HR
2	规范和设计的动画演示	C.5.26	R	R	R	R
3	静态分析	B.6.4 表 B.8	R	HR	HR	HR
4	动态分析和测试	B.6.5 表 B.2	R	HR	HR	HR
5	软件设计规范和软件验证（包括数据验证）计划之间向前可追溯	C.2.11	R	R	HR	HR
6	软件验证（包括数据验证）计划和软件设计规范之间向后可追溯	C.2.11	R	R	HR	HR
7	离线数值分析	C.2.13	R	R	HR	HR
软件模块测试和集成		见表 A.5				
可编程电子集成测试		见表 A.6				
软件系统测试（确认）		见表 A.7				

注1：为方便起见，所有的验证活动已经在表中罗列。然而，本表不对表A.5和A.6（同样是验证活动）中验证的动态测试方法提出附加的要求。也不对除了软件确认（见表A.7）以外验证测试提出要求，在本标准中软件确认是用来证明对安全要求规范的符合性（端到端验证）。

注2：验证活动贯穿GB/T20438.1、GB/T20438.2和GB/T20438.3，因此安全相关系统的首次验证是针对前期系统层面的规范。

注3：在软件安全生命周期的早期阶段，验证是静态的，例如通过检查、复审和形式化证明。当产生代码后可进行动态测试。验证需要两类信息的综合。如通过静态方法对软件模块的代码验证包括软件检查、走查、静态分析、形式化证明等技术，动态方法的代码验证包括功能测试、白盒测试、统计测试。通过两类证据的结合，证明每一软件模块满足有关规范。

注4：见表C.9。

注5：第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。

表A.10　功能安全评估（见第6章）

技术/措施[a]		Ref	SIL1	SIL2	SIL3	SIL4
1	检查表	B.2.5	R	R	R	R
2	判定/真值表	C.6.1	R	R	R	R
3	失效分析	表B4	R	R	HR	HR
4	多样化软件的共因失效分析（如实际使用了多样化软件）	C.6.3	---	R	HR	HR
5	可靠性框图	C.6.5	R	R	R	R
6	第8章要求和软件功能安全评估计划之间的向前可追溯	C.2.11	R	R	HR	HR
注1：见表C.10。 **注2：**第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。						

附 录 B
（资料性附录）
详细表格

表B.1 设计和编码标准

（被表 A.4 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	使用编码标准以降低错误发生的可能性	C.2.6.2	HR	HR	HR	HR
2	无动态对象	C.2.6.3	R	HR	HR	HR
3a	无动态变量	C.2.6.3	---	R	HR	HR
3b	动态变量加载的在线检查	C.2.6.4	---	R	HR	HR
4	有限制的使用中断	C.2.6.5	R	R	HR	HR
5	有限制的使用指针	C.2.6.6	---	R	HR	HR
6	有限制的使用递归	C.2.6.7	---	R	HR	HR
7	在高级语言程序中无非结构化控制流	C.2.6.2	R	HR	HR	HR
8	无自动类型转换	C.2.6.2	R	HR	HR	HR

注1：措施2、3a和5使用动态对象(例如执行堆栈或堆)可能会对可用内存和执行时间上施加要求。在以下情况下，不需要应用措施2、3a和5，即如果使用编译器能够确保：a）在运行前，为所有的动态变量和对象分配了足够的内存，或者保证一旦发生内存分配错误，能够实现安全状态；b)其响应时间满足要求。

注2：见表C.11。

注3：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表B.2 动态分析和测试

（被表 A.5 和 A.9 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	根据边界值分析执行测试用例	C.5.4	R	HR	HR	HR
2	根据错误推测执行测试用例	C.5.5	R	R	R	R
3	根据错误植入执行测试用例	C.5.6	---	R	R	R
4	根据基于模型测试用例的生成执行测试用例	C.5.27	R	R	HR	HR
5	性能建模	C.5.20	R	R	R	HR
6	等价类和输入划分测试	C.5.7	R	R	R	HR
7a	结构测试覆盖率(入口)100% [b]	C.5.8	HR	HR	HR	HR
7b	结构测试覆盖率(语句)100% [b]	C.5.8	R	HR	HR	HR
7c	结构测试覆盖率(分支)100% [b]	C.5.8	R	R	HR	HR
7d	结构测试覆盖率(条件、MC/DC)100% [b]	C.5.8	R	R	R	HR

注1：测试用例分析在子系统级进行并基于规范和/或规范和代码。 **注2：**见表C.12。 **注3：**第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。
[a] 应根据安全完整性等级选择适当的技术/措施。 [b] 当100%覆盖率不能实现时（比如防御性代码的语句覆盖率），应给予适当的说明。

表B.3　功能和黑盒测试

（被表A.5、A.6和A.7引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	根据因果图执行测试用例	B.6.6.2	---	---	R	R
2	根据基于模型测试用例的生成执行测试用例	C.5.27	R	R	HR	HR
3	原型设计/动画	C.5.17	---	---	R	HR
4	等价类和输入划分测试，包括边界值分析	C.5.7 C.5.4	R	HR	HR	HR
5	过程仿真	C.5.18	R	R	R	R
注1：测试用例分析在软件系统级进行并只基于规范。 **注2：**仿真的完整性取决于安全完整性等级、复杂性和应用。 **注3：**见表C.13。 **注4：**第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。						

表B.4　失效分析

（被表A.10引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1a	因果图	B.6.6.2	R	R	R	R
1b	事件树分析	B.6.6.3	R	R	R	R
2	故障树分析	B.6.6.5	R	R	R	R
3	软件功能失效分析	B.6.6.4	R	R	R	R
注1：为了将软件分类到最合适的软件完整性等级，应进行预先危险分析。 **注2：**见表C.14 。 **注3：**第三列中的参考（属于资料性的，而非规范性的）“B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录C中给出的属性进行论证，以满足特定的应用。						

表B.5　建模

（被表 A.7 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	数据流图	C.2.2	R	R	R	R
2a	有限状态机	B.2.3.2	---	R	HR	HR
2b	形式化方法	B.2.2，C.2.4	---	R	R	HR
2c	时间佩特里（Petri）网	B.2.3.3	---	R	HR	HR
3	性能建模	C.5.20	R	HR	HR	HR
4	原型设计/动画	C.5.17	R	R	R	R
5	结构图	C.2.3	R	R	R	HR

注1：未在表中列出的特定技术也可以在考虑范围之内，且应符合本标准。

注2：不要求量化概率。

注3：见表C.15 。

注4：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表B.6　性能测试

（被表 A.5 和 A.6 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	雪崩/压力测试	C.5.21	R	R	HR	HR
2	响应时间和存储约束	C.5.22	HR	HR	HR	HR
3	性能要求	C.5.19	HR	HR	HR	HR

注1：见表C.16 。

注2：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。

表B.7　半形式化方法

（被表 A.1、A.2 和 A.4 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	逻辑/功能块图	见下注 1	R	R	HR	HR
2	时序图	见下注 1	R	R	HR	HR
3	数据流图	C.2.2	R	R	R	R
4a	有限状态机/状态转移图	B.2.3.2	R	R	HR	HR
4b	时间佩特里（Petris）网	B.2.3.3	R	R	HR	HR
5	实体-关系-属性数据模型	B.2.4.4	R	R	R	R
6	消息序列图	C.2.14	R	R	R	R

7	判定/真值表	C.6.1	R	R	HR	HR
8	统一建模语言（UML）	C.3.12	R	R	R	R

注1：逻辑/功能块图和时序图在GB/T 15969.3中有描述。

注2：见表C.17 。

注3：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表B.8 静态分析

（被表 A.9 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	边界值分析	C.5.4	R	R	HR	HR
2	检查表	B.2.5	R	R	R	R
3	控制流分析	C.5.9	R	HR	HR	HR
4	数据流分析	C.5.10	R	HR	HR	HR
5	错误推测	C.5.5	R	R	R	R
6a	形式化检查，包括具体的准则	C.5.14	R	R	HR	HR
6b	走查(软件)	C.5.15	R	R	R	R
7	符号执行	C.5.11	---	---	R	R
8	设计复审	C.5.16	HR	HR	HR	HR
9	运行时错误行为的静态分析	B.2.2 ，C.2.4	R	R	R	R
10	最坏情况的执行时间分析	C.5.20	R	R	R	R

注1：见表C.18 。

注2：第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。

[a] 应根据安全完整性等级选择适当的技术/措施。可替代的或等价的技术/措施用字母紧随数字的方式予以表示。其目的是只需选取可替代的或等价的技术/措施中的一种。替代技术的选择应该根据附录 C 中给出的属性进行论证，以满足特定的应用。

表B.9　模块化方法

（被表 A.4 引用）

技术/措施[a]		参考	SIL1	SIL2	SIL3	SIL4
1	软件模块规模限制	C.2.9	HR	HR	HR	HR
2	软件复杂度控制	C.5.13	R	R	HR	HR
3	信息隐藏/封装	C.2.8	R	HR	HR	HR
4	参数数量限制/固定数量的子程序参数	C.2.9	R	R	R	R
5	子程序和函数的单入口/单出口	C.2.9	HR	HR	HR	HR
6	充分定义的接口	C.2.9	HR	HR	HR	HR
注1：见表C.19 。 **注2：**第三列中的参考（属于资料性的，而非规范性的） “B.x.x.x”，“C.x.x.x”显示了GB/T20438.7中附录B和附录C给出的技术/措施的详细描述。						
[a] 应根据安全完整性等级选择适当的技术/措施。没有一个单一的技术是充分的，应考虑所有适当的技术。						

附 录 C
（资料性附录）
软件系统性能力的属性

C.1 概述

由于影响软件系统性能力的因素有很多，不可能给出一个能符合所有给定应用的技术和措施组合的算法。附录C的目的是：

——为从附录 A 和 B 选择具体技术来实现软件系统性能力提供指导。

——描述了论证使用未明确地在附录 A 和 B 中所列技术的基本方法。

附录C是附录A和B表的补充。

C.1.1 与附录A和B相关联的附录C的结构

在表1中定义了软件安全生命周期每个阶段的输出。例如，考虑软件安全要求规范。

附录A的表A.1（“软件安全要求规范”）为编制软件安全要求规范推荐了具体的技术。

技术/措施		参考	SIL1	SIL2	SIL3	SIL4
1a	半形式化方法	表 B.7	R	R	HR	HR
1b	形式化方法	B.2.2，C.2.4	---	R	R	HR
2	在系统安全要求和软件安全要求间向前可追溯	C.2.11	R	R	HR	HR
3	在安全要求和认识到的安全需求间向后可追溯	C.2.11	R	R	HR	HR
4	支持上述适当的技术或措施的计算机辅助规范工具	B.2.4	R	R	HR	HR

附录C的表C.1(“系统性安全完整性属性---软件安全要求规范”)指出，软件安全要求规范是用以下期望属性来表征的（这是在GB/T20438.7的附录F中非正式定义的）：

属性					
针对由软件来确保的安全需求的完整性	针对由软件来确保的安全需求的正确性	没有规范本身的错误，包括：语义含糊	安全要求的可理解性	针对由软件来确保的安全需求，没有非安全功能的不利干扰	为验证和确认提供基础的能力

附录C的表C.1也对具体技术在实现这些期望属性的有效性上做了一个非正式的分级 R1/R2/R3。

技术/措施		属性					
		针对由软件来确保的安全需求的完整性	针对由软件来确保的安全需求的正确性	没有规范本身的错误，包括：语义含糊	安全要求的可理解性	针对由软件来确保的安全需求，没有非安全功能的不利干扰	为验证和确认提供基础的能力
1a	半形式化方法	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号 R2 根据覆盖率准则验证规范	R1 方法和符号，有助于避免或检测内部不一致性、遗漏的行为或数学上的不一致的表述 R2 根据覆盖率准则验证规范 R3 基于系统性分析规范的验证，和/或对特定类型的规范本身故障的系统性避免	R1 已定义的符号，以减少误解 R2 规范中复杂度限制的应用	——	R2 已定义的符号，以减少规范中的歧义

把软件安全要求规范作为安全软件基础的置信度，取决于所用技术的严格性，运用该技术已经实现了软件安全要求规范的期望属性。技术的严格性是由R1至R3的范围进行划分。R1严格性最低，R3严格性最高。

R1	无客观可接受准则，或有限的客观可接受准则。例如，基于判断、现场试验的黑盒测试。
R2	有客观可接受的准则能对实现要求的属性给出高置信度（需识别和论证例外）；例如，有覆盖指标、检查表覆盖率的测试或分析技术。
R3	有客观的、系统的推理能证明实现了要求的属性。例如形式化证明、证明符合的架构约束能够保证此属性。
—	这种技术与此属性不相关。

技术也许能实现R1/R2/R3等级中的一种，这与特定属性有关，取决于技术满足的严格等级。

技术/措施		属性					
		针对由软件来确保的安全需求的完整性	针对由软件来确保的安全需求的正确性	没有规范本身的错误，包括：语义含糊	安全要求的可理解性	针对由软件来确保的安全需求，没有非安全功能的不利干扰	为验证和确认提供基础的能力
1a	半形式化方法				R1 已定义的符号，以减少误解 R2 规范中复杂度限制的应用		

在这个例子中，半形式化的方法通过采用一种受限的符号来提高表达的准确性达到了R1严格度，并且通过进一步限制规范的复杂度达到了R2，否则可能产生混淆。

C.1.2 使用方法 - 1

出于指导目的，如果能确切地证明要求的属性已经在软件安全要求规范编制过程中得以实现，则已经论证了以下置信度，即软件安全要求规范是开发有足够系统性安全完整性软件的充分基础。

附录C的表C.1指出，附录A表A.1中每个技术在不同程度上实现一个或多个上述表C.1中的与软件安全要求规范相关的属性。

然而，重要的是要注意，虽然附录A的表A.1推荐了具体的技术，但是这些建议不是一种规定，并且实际上附录A明确表示“考虑到大量影响软件系统性能力的因素，不可能给出一种将技术和措施结合起来的、对任何指定的应用都正确的算法”。

实践中，编制软件安全要求规范时使用技术的选择，除了受这些技术本身的性能影响外，还易受到一些实际的限制（见7.1.2.7）。这样的限制可能包括：

——为整个开发周期选择的方法、语言和工具的一致性和互补性；

——开发者是否完全理解使用方法、语言和工具；

——在开发期间，方法、语言和工具是否对遇到特定问题有很好的适用性。

在实现软件安全要求规范生命周期的要求属性中，同时也考虑在特定的开发项目中的实际约束，可以使用表C.1来与附录A表A.1具体技术的有效性进行相对的比较。

例如，一种形式化方法（R3）比一种半形式化方法（R2）能够提供一个更好的验证和确认依据，但是其他的项目约束（比如复杂的计算机支持工具的可用性，或形式化符号的非常专业的表达能力）也许更适合选一种半形式化方法。

这样，表C.1的期望属性可以为附录A表A.1推荐的用于开发软件安全要求规范的可选技术的合理和可行的比较提供依据。或者更通俗的说，可以通过考虑附录C相应表中所列的期望属性，从附录A推荐的用于特定生命周期阶段的若干可选技术中做合理的选择。

但是，需要仔细注意的是，由于系统性行为的性质，附录C的这些属性可能无法在最严格的条件下实现或证明。相反，它们是需要满足的目标。它们的实现甚至可能必须在不同的属性间做权衡，例如在防御性设计和简单性设计之间。

最后，除了定义R1/R2/R3准则，出于指导的目的，在如下两者之间建立非正式的联系也是非常有用的：（1）从R1到R3严格等级递增和（2）软件正确性的置信度递增。作为一般并且非正式的建议，当附录A要求相应的SIL性能时，应以下面最低限度的严格等级为目标：

SIL	严格度 R
1/2	R1
3	R2，如果可用
4	可用的最高严格度

C.1.3 使用方法 - 2

尽管附录A推荐了具体的技术，只要满足生命周期阶段的要求和目标，也允许应用其他措施和技术。

前面已经指出影响软件系统性能力的因素有很多，并且不可能给出一种算法，通过选择和组合技术来保证在任何给定应用中实现期望的属性。

可能会有几种有效的方法来实现期望的属性，并且应该认识到，系统开发者也许能够提供可选的证据。为了论证不同于附录A的技术选择，可以使用附录C中的信息作为合理性判断的基础。

C.2 系统性安全完整性的属性

本指导和GB/T 20438.7中指明了实现系统性安全完整性属性和形成可信证据的具体技术。如果某种方法对实现某个属性没有贡献，在下表中用一个破折号表示。如果某种方法对一些属性有负面影响，而对其他方面有积极影响，在下面的相关表中提供了注释。

表 C.1　系统性安全完整性的属性—软件安全要求规范

（见 7.2，被表 A.1 引用）

技术/措施		属性					
		针对由软件来确保的安全需求的完整性	针对由软件来确保的安全需求的正确性	没有规范本身的错误，包括：语义含糊	安全要求的可理解性	针对由软件来确保的安全需求，没有非安全功能的不利干扰	为验证和确认提供基础的能力
1a	半形式化方法	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号 R2 根据覆盖率准则验证规范	R1 方法和符号，有助于避免或检测内部不一致性、遗漏的行为或数学上的不一致的表述 R2 根据覆盖率准则验证规范 R3 基于系统性分析规范的验证，和/或对特定类型的规范本身故障的系统性避免	R1 已定义的符号，以减少误解 R2 规范中复杂度限制的应用	——	R2 已定义的符号，以减少规范中的歧义
1b	形式化方法	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号	R1 被领域专家使用的应用友好的或领域专用的规范方法和符号 R2 根据覆盖率	R1 方法和符号，有助于避免或检测内部不一致性、遗漏的行为或数学上的不一致的表述 R2 根据覆盖率	—— 注：如果方法不是应用友好的或领域专用的，那么可能使这个属性的实现复杂化	——	R3 减少规范中的歧义

			准则验证规范 R3 保证行为限定方面的正确性	准则验证规范 R3 基于系统性分析规范的验证，和/或对特定类型的规范本身故障的系统性避免			
2	在系统安全要求规范和软件安全要求间向前可追溯	R1 软件安全要求规范可满足系统安全要求的置信度	——	——	——	——	——
3	在软件安全要求规范和获知到的安全要求间向后可追溯	——	R1 软件安全要求规范未包含不必要的复杂性的置信度	——	R1 对 EUC 安全需求的可追溯增强了可理解性	R1	R1
4	支持上述适当的技术或措施的计算机辅助规范工具	R1 EUC 和软件环境的领域知识的封装 R2 需考虑事项的检查表是否已定义、已论证和已全面覆盖	R1 功能仿真技术 R2 根据已定义的和已论证的覆盖准则的功能仿真	R2 语义和语法检查，以确保符合相关规则	R1 规范的动画演示或浏览	R1 安全和非安全功能识别	R1 辅助可追溯性和覆盖 R2 可追溯性和覆盖率的度量

表 C.2　系统性安全完整性的属性—软件设计和开发—软件架构设计

（见 7.4.3，被表 A.2 引用）

技术/措施		属性							
		针对软件安全要求规范的完整性	针对软件安全要求规范的正确性	没有本身的设计错误	简单性和易懂性	行为的可预测性	可验证和可测试的设计	故障裕度	抵御外部事件造成的共因失效
1	故障检测	——	——	——	—— 注：可能使这个属性的实现复杂化	R1 逻辑程序流监视提供可预测性	——	R1 （如果覆盖目标是已定义、已论证和已满足要求的，则 R2）	R1 或 ——
2	错误检测代码	——	——	——	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	——	R1 （如果覆盖目标是已定义、已论证和已满足要求的，则 R2） 对特定应用领域有效，例如，数据通信	R1 对特定应用领域有效，例如，数据通信
3a	失效断言编程	——	R2 后断言可以检查与详细要求的符合性	——	R2 预断言限制输入空间	R2 后断言检查期望的/可接受的输出	R2 预断言限制输入空间，因而限制所需的测试空间	R3 对目标失效有效	R3 对目标失效有效
3b	多样化监视技术(同一台计算机上的监视功	——	——	R2 多样化监视只实现了最低的安全要求	R2 多样化监视提供了隐式多样性	R2 多样化监视以一种简单的方式只实现了最低的安全要求	R2 多样化监视只实现了最低的安全要求	R1 （如果覆盖目标是已定义、已论证和已满足要	R1 （如果覆盖目标是已定义、已论证和已满足要求

	能和被监视功能之间独立）							求的，则 R2）	的，则 R2）
3c	多样化监视技术（监视计算机与被监视计算机之间分离）	——	——	R2 多样化监视只实现了最低的安全要求	R2 多样化监视提供了隐式多样性	R2 多样化监视只实现了最低的安全要求	R2 多样化监视只实现了最低的安全要求	R1 （如果覆盖目标是已定义、已论证和已满足要求的，则 R2）	R1 （如果覆盖目标是已定义、已论证和已满足要求的，则 R2）
3d	多样化冗余，实现相同软件安全要求规范	——	——	——	—— 注：如果在相同的可执行软件里完成，本属性的实现可能复杂化	——	——	R1 如果一个程序的失效不会对其他程序产生负面影响 R2 如果覆盖目标是已定义、已论证和已满足要求 不能防止要求规范的故障	R1 如果一个程序的失效不会对其他程序产生负面影响 R2 如果覆盖目标是已定义、已论证和已满足要求 不能防止要求规范的故障
3e	功能上多样化冗余，实现不同软件安全要求规范。这通常要求传感器在不同的物理原理下运行。	——	——	R1	—— 注：如果在相同的可执行软件里完成，本属性的实现可能复杂化	——	——	R1 如果一个程序的失效不会对其他程序产生负面影响	R1 如果一个程序的失效不会对其他程序产生负面影响。 防止规范故障

3f	后向恢复	——	——	—— 注：可能使这个属性的实现复杂化	——	—— 注：可能使这个属性的实现复杂化	——	R2	R1 （如果覆盖目标是已定义、已论证和已满足要求，则 R2）
3g	无状态设计（或者有限状态设计）	R2 如果安全要求也是无状态的或有限状态	R2 如果安全要求也是无状态的或有限状态	R2 如果安全要求也是无状态的或有限状态	R1 R2 对于可能的状态，定义、论证并且满足了数目的限制	R1 R2 对于可能的状态，定义、论证并且满足了数目的限制	R1 R2 针对可能状态的验证/测试覆盖，如果目标已定义、已论证并且满足	R1 如果这导致自我修复的设计 R2 针对自我修复，如果目标是已定义、已论证并且满足	R1 如果这导致自我修复的设计 R2 针对自我修复，如果目标是已定义、已论证并且满足
4a	故障恢复重试机制	——	——	——	——	—— 注：可能使这个属性的实现复杂化	——	R1 （如果覆盖目标是已定义、已论证和已满足要求，则 R2）	R1 （如果覆盖目标是已定义、已论证和已满足要求，则 R2）
4b	适度降级	——	——	—— 注：可能使这个属性的实现复杂化	——	——	——	R1 R2 如果覆盖目标是已定义、已论证和已满足要求	R1 R2 如果覆盖目标是已定义、已论证和已满足要求
5	人工智能——故障纠正	——	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	——	——
6	动态再配置	——	—— 注：可能使这个属性的实现	—— 注：可能使这个属性的实现	—— 注：可能使这个属性的实现	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现	——	——

			复杂化	复杂化	复杂化		复杂化		
7	模块化方法	——	R1 R2 如果覆盖目标是已定义、已论证和已满足要求，则 R2。否则，只能达到 R1.	R1 如果对每个模块可独立地验证没有特定类型的本身设计故障 R3 如果有基于模块化设计的严格理由支持没有特定类型的本身设计故障	R1 R2 如果模块化的目标是已定义、已论证并且满足要求	R1 R2 如果模块化的目标是已定义、已论证并且满足要求	R1 R2 如果模块化的目标是已定义、已论证并且满足要求	R1 如果不受某个模块失效影响的那些模块有助于缓解/恢复 R3 如果对特定故障的容忍有严格的理由支持	R1 如果受外部事件（这些事件可同时影响多通道）影响的模块可被识别并彻底的验证 R3 如果对特定外部事件的容忍有严格的理由支持
8	使用可信赖/已证实的软件模块和组件(如可获得)	——	R1 R2 R3 如果组件十分有助于特定的安全要求，并被正确的使用	R1 R2 R3 复用经证明的组件。对于组件的这种能力应该被论证	R1 模块化方法把整体复杂的模块分解成可理解的单元	R1 R2 R3 复用经过证明的组件	——	R1 R2 如果组件很容易提供故障裕度能力并被正确的使用，或者在组件周围建立了一个故障裕度层，则 R2	R1 R2 如果组件很容易对能同时影响多个通道的可能外部事件进行防御，并被正确的使用，或者在组件周围建立了一个防御层，则 R2
9	在软件安全要求规范和软件架构间向前可追溯	R1 确信架构覆盖了软件安全要求	——	——	——	——	——	——	——
10	在软件安全要	——	R1 确信架构	——	——	——	——	——	——

	求规范和软件架构间向后可追溯		不包含不必要的复杂性						
11a	结构图表的方法	——	R1	——	R1 （图形化的描述更容易理解）	——	R1 （结构化的设计更容易验证和测试）	——	——
11b	半形式化方法	R1 一种应用友好的或领域专用的规范方法和符号	R1 一种应用友好的或领域专用的规范方法和符号	R3 可以检测内部的不一致性或缺失的行为或数学上的不一致表达	——	R2 （提供可预测性的证据）	R2 （提供设计模型的内部一致性证据）	——	——
11c	形式化设计和优化方法	R1 一种应用友好的或领域专用的规范方法和符号	R1 提供应适用于该领域行为限制方面的精确定义	R3 可以检测内部的不一致性或缺失的行为或数学上的不一致表达	—— 注：可能使这个属性的实现复杂化	R2 提供可预测性的证据	R2	——	——
11d	自动软件生成	R1 如果可执行的软件是根据要求规范自动产生的，或者来自一个已被证明是完整的设计 R2 如果生成工具有适当的系谱	R1 如果可执行的软件是根据要求规范自动产生的，或者来自一个已被证明是完整的设计 R2 如果生成工具有适当的系谱	R1 如果生成工具保证避免特定的固有设计故障 R2 如果生成工具有适当的系谱	——	——	——	R1 R2 R3 如果故障裕度能力是自动生成的	——

12	计算机辅助规范和设计工具	R1 EUC 和软件环境的领域知识的封装 R2 需考虑事项的检查表是否已定义、已论证和已全面覆盖	R1 加强后向要求跟踪 功能仿真技术 R2 依据明确和合理的覆盖标准的功能仿真	R2 语义和语法检查，以确保符合相关规则	R1 动画和浏览	——	R2 语义和语法检查，以确保符合相关规则	——	——
13a	周期性运转，并且确定最大周期时间	——	R1 针对规范的时间因素 R3 如果通过严格的推理确定最大周期时间	R1 针对规范的时间因素 R3 如果通过严格的推理确定最大周期时间	——	R1 针对规范的时间因素 R3 如果通过严格的推理确定最大周期时间	R1 针对规范的时间因素 R3 如果通过严格的推理确定最大周期时间	——	——
13b	时间触发式架构	R3 完整性由分配保证（仅适用于时间属性）	R3 正确性由分配保证（仅适用于时间属性）	R3 严格保证不发生本身的时间故障	R1 作为方法可预见，定义的符号可显著减少误解	R3 不良干扰：时域完全分离，无干扰	R3 显著减少了测试和认证系统所需的工作	R2 故障裕度的明确实施	R3 外部中断不能干扰时间触发计划，以给安全关键的任务优先级
13c	事件驱动，并且确保最大响应时间	——	——	——	R1 事件驱动的架构可能会阻碍可理解性	R2 事件驱动的架构可能会阻碍可理解性	R1 使测试更易预测	——	——
14	静态资源分配	R1	R1	R1	R1 使设计更易理解	R2 在架构中定义资源的使用	R1 使测试更易预测	——	——

15	访问共享资源的静态同步	——	R1 资源访问中给予了可预测性	R1 如果同步的正确性经过严格推理，则R3	R1 使设计更易理解	R1 如果同步的正确性经过严格推理，则R3	——	——	——

表C.3　系统性安全完整性的属性—软件设计和开发—支持工具和编程语言

（见7.4.4，被表A.3引用）

技术/措施		属性		
		工具对具有所需软件属性的软件生成的支持程度	工具的操作和功能的清晰度	输出的正确性和可重复性
1	适当的编程语言	R2 如果是强类型，限制类型转换 R3 如果是为严格的推理而定义的语义	——	——
2	强类型编程语言	R2	——	——
3	语言子集	R2 取决于选择的子集	R1	R2 取决于选择的子集
4a	已认证的工具	——	R2	R2
4b	工具：通过使用提高置信度	R1 如果对检测到的程序错误的分类进行了系统性定义 R2 如果对于工具的性能有客观的确认证据	R1 如果工具支持不是专用于某个问题领域 R2 如果工具支持明显是专用于某个问题领域	R1 R2 如果对于工具的性能有客观的确认证据，例如一个编译器确认套件

表 C.4　系统性安全完整性的属性—软件设计和开发—详细设计

（包括软件系统设计、软件模块设计和编码）

（见 7.4.5 和 7.4.6，被表 A.4 引用）

技术/措施		属性							
		针对软件安全要求规范的完整性；	针对软件安全要求规范的正确性；	防止固有设计故障；	简单性和易懂性；	行为可预测性；	可验证和可测试的设计；	故障裕度/故障检测；	防止共因失效。
1a	结构化方法	R2	R1	R1	——	——	R1 结构化设计更容易验证和测试	——	——
1b	1b 半形式化方法	R2	R2	R2	——	R2	R2	——	——
1c	1c 形式化设计和优化方法	——	R3	R3	—— 注：可能使这个属性的实现复杂化	R3 为可预测性提供证据	R2	——	——
2	计算机辅助设计工具	R2 取决于计算机辅助规范工具运用语义和语法检查，以确保满足相关规则	R1	R2 取决于计算机辅助规范工具运用语义和语法检查，以确保满足相关规则	——	——	R2 取决于 CASE 工具，以支持测试覆盖率和静态验证	——	——
3	防御性编程	——	——	——	—— 注：可能使这个属性的实现复杂化	—— 注：可能使这个属性的实现复杂化	——	R1 （如果覆盖目标是已定义、已论证和已满足要求，则 R2）	R1 （如果覆盖目标是已定义、已论证和已满足要求，则 R2）
4	模块化方法	——	——	R1	R1	R1	R1	——	——

5	设计和编码标准	——	——	R1	R1	R1	R1	——	——
6	结构化编程	——	R1	R1	R1	R1	R1	——	——
7	使用可信的/经验证的软件组件（如可获得）	——	——	R1 复用已证明的组件	R1 模块化方法把整体复杂性分解成可以理解的单元	R1 组件的行为是已知的	——	——	——
8	在软件安全要求规范和软件设计间向前可追溯	R1 确信设计满足软件安全要求	——	——	——	——	——	——	——

表 C.5　系统性安全完整性的属性—软件设计和开发—软件模块测试和集成

（见 7.4.7 和 7.4.8，被表 A.5 引用）

技术/措施		属性			
		针对软件设计规范，所做测试的完整性	针对软件设计规范（成功完成），所做测试的正确性	可重复性	精确定义的测试配置
1	概率测试	R1 （如果操作剖面覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
2	动态分析和测试	R1 （如果结构化覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
3	数据记录和分析	——	R1	R1 在测试规程中提升一致性	R2 如果故障记录/测试日志包含软件基线的细节
4	功能和黑盒测试	R1 （如果操作剖面覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
5	性能测试	——	R1	——	——

			（如果要求的输出是已定义、已论证和满足要求的，则 R2）		
6	基于模型的测试（MBT）	R2 MBT 能提前暴露在规范和设计中的歧义，MBT 过程从要求开始 R3 如果严格的推理应用于建模，并且使用了测试用例生成（TCG）	R2 结果评估和回归测试套件是 MBT 的一个主要获益点 R3 如果运用严格的建模方法，那么客观的覆盖证据是可能的	R3 MBT（带 TCG）旨在生成测试的自动执行	R2 MBT 是自动的，测试配置要精确定义；生成的测试的执行和黑盒测试类似，这些黑盒测试具有与源代码级覆盖测量结合的可能性
7	接口测试	——	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
8	测试管理和自动化工具	R1 （如果测试覆盖目标是已定义、已论证和满足要求的，则 R2）	——	R1 自动化提升一致性	R2 提供了测试的可重复性
9	在软件设计规范与模块及集成测试规范间向前可追溯	R1 确信测试规范满足软件安全要求	——	——	R2 确信待测的要求具有清晰的基线
10	形式化验证	R3 如果严格的推理应用于测试用例的构建，以说明设计的所有方面已经实施	R3 提供了满足所有软件安全要求的客观依据	R1 如果没有可用的支持工具 R2 如果有支持的工具	——

表 C.6　系统性安全完整性的属性—可编程电子集成（硬件和软件）

（见 7.5.　被表 A.6 引用）

技术/措施		属性			
		针对设计规范，所做集成的完整性	针对设计规范（成功完成），所做集成的正确性	可重复性	精确定义的集成配置
1	功能和黑盒测试	R1 （如果操作剖面覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——

2	性能测试	——	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
3	在系统及针对硬件/软件集成的软件设计要求和硬件/软件集成测试规范之间向前可追溯	R1 确信硬件/软件集成测试规范满足集成要求	——	——	R2 确信待测的要求具有清晰的基线

表 C.7　系统性安全完整性的属性——系统安全确认的软件方面

（见 7.7，被表 A.7 引用）

技术/措施		属性			
		针对软件设计规范，确认的完整性	针对软件设计规范（成功完成），确认的正确性	可重复性	精确定义的确认配置
1	概率测试	R1 （如果操作剖面覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则R2）	——	-
2	过程仿真	R1	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	R2 提供了一个外部环境的定义
3	功能和黑盒测试	R1 （如果操作剖面覆盖目标是已定义、已论证和满足要求的，则 R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	-
4	在软件安全要求规范和软件安全确认计划间的向前可追溯	R1 确信软件安全确认计划满足软件安全要求	——	——	R2 确信待测的要求具有清晰的基线

5	在软件安全确认计划和软件安全要求规范间的向后可追溯	——	R1 确信软件安全确认计划不包含不必要的复杂性	——	R2 确信待测的要求具有清晰的基线

表 C.8　系统性安全完整性属性—软件修改

（见 7.8，被表 A.8 引用）

技术/措施		属性					
		针对要求,修改的完整性	针对要求，修改的正确性	防止引入固有的设计故障	避免不需要的行为	可验证和可测试的设计	回归测试和验证覆盖
1	影响分析	——	——	——	R1	R1	R1
2	再验证被变更的软件模块	R1 （如果有客观验证的目标,则 R2）	R1 （如果有客观验证的目标，则 R2）	R1 （如果有客观验证的目标,则R2）	——	——	R1 （如果有客观验证的目标，则 R2）
3	再验证受影响的软件模块	R1 （如果有客观验证的目标,则 R2）	R1 （如果有客观验证的目标，则 R2）	R1 （如果有客观验证的目标,则R2）	——	——	R1 （如果有客观验证的目标，则 R2）
4a	再确认整个系统	R1 （如果有客观验证的目标,则 R2）	R1 （如果有客观验证的目标，则 R2）	——	R1 （如果有客观验证的目标，则 R2）	——	R1 （如果有客观验证的目标，则 R2）
4b	回归确认	R1 （如果有客观验证的目标,则 R2）	R1 （如果有客观验证的目标，则 R2）	——	R1 （如果有客观验证的目标，则 R2）	——	R1 （如果有客观验证的目标，则 R2）
5	软件配置管理	——	——	——	——	——	R1
6	数据记录和分析	R1	R1	——	——	——	——

7	在软件安全要求规范和软件修改计划（包括再验证和再确认）之间的向前可追溯	R1 确信软件修改计划(包括再验证和再确认)满足软件安全要求	——	——	——	——	——
8	软件修改计划（包括再验证和再确认）和软件安全要求规范之间的向后可追溯	——	R1 确信软件修改计划（包括再验证和再确认）不包含不必要的复杂性	——	——	——	——

表 C.9　系统性安全完整性的属性—软件验证

（见 7.9，被表 A.9 引用）

技术/措施		属性			
		针对前一阶段，验证的完整性	针对前一阶段（成功完成），验证的正确性	可重复性	精确定义的验证配置
1	形式化证明	——	R3	——	——
2	规范和设计的动画演示	R1	R1	——	——
3	静态分析	——	R1/R2/R3 （严格度的范围可以从语言子集的强制使用到数学形式化分析）	——	——
4	动态分析和测试	R1 （如果结构化覆盖目标是已定义、已论证和满足要求的，则R2）	R1 （如果要求的输出是已定义、已论证和满足要求的，则 R2）	——	——
5	软件设计规范和软件验证(包括数据验证)计划之间的向前可追溯	R1 确信软件验证（包括数据验证）计划满足软件安全要求	——	——	R2 确信待测的要求具有清晰的基线

6	软件验证(包括数据验证)计划和软件设计规范之间的向后可追溯	——	R1 确信软件验证（包括数据验证）计划不包含不必要的复杂性	——	R2 确信待测的要求具有清晰的基线
7	离线数值分析	——	R1 提高经过很好计算的期望数值精度的置信度 （带客观可接受准则，则R2。如果可接受准则经过客观系统性推理，则R3）	——	——

表 C.10　系统性安全完整性的属性--功能安全评估

（见第 8 章，被表 A.10 引用）

技术/措施		属性						
		针对本标准的功能安全评估的完整性	针对设计规范（成功实施），功能安全评估的正确性	所有被识别问题的可追溯的闭环	不需要大量的重新评估的变更后，修改功能安全评估的能力	可重复性	及时性	精确定义的配置
1	检查表	R1	R1	R1	-	R1	——	——
2	判定/真值表	R1	R2	——	——	R2	——	——
3	失效分析	R2	R2	R1 （失效分析基于已认可的失效列表）	——	R1 （失效分析基于已认可的失效列表）	——	——
4	多样化软件的共因失效分析（如实际使用了多样化软件）	R2	R2	R1 （如果CCF分析基于已认可的CC始发列表）	——	R1 （如果CCF分析基于已认可的CC始发列表）	——	——
5	可靠性框图	R1	R1	——	——	——	——	——

6	本部分第 8 章要求和软件功能安全评估计划之间的向前可追溯	R1 确信软件功能安全评估计划满足本部分第8章的要求	——	——	——	——	——	——

C.3 系统性安全完整性的属性—详细表格

表 C.11 详细属性—设计和编码标准

（被表 B.1 引用）

技术/措施		属性							
		针对软件安全要求，规范的完整性；	针对软件安全要求，规范的正确性；	防止固有设计故障；	简单性和易懂性；	行为可预测性；	可验证和可测试的设计；	故障裕度/故障检测；	防止共因失效。
1	使用编码标准以降低错误发生的可能性	——	——	R1	R1 消除已选语言结构	R1	R1	——	——
2	无动态对象	——	——	R1/R2/R3 取决于使用的语言	——	R1/R2/R3 取决于使用的语言	R1/R2 取决于使用的语言	——	——
3a	无动态变量	——	——	R1/R2/R3 取决于使用的语言	——	R1/R2/R3 取决于使用的语言	R1/R2 取决于使用的语言	——	——
3b	动态变量安装的在线检查	——	——	R1/R2/R3 取决于使用的语言	——	R1/R2/R3 取决于使用的语言	R1/R2 取决于使用的语言	——	——
4	有限制的使用中断	——	——	R1/R2 取决于使用的语言	R1 增加逻辑和事件序列的清晰	R1/R2 取决于使用的语言	R1/R2 取决于使用的语言	——	——

					度				
5	有限制的使用指针	——	——	R1/R2 取决于使用的语言	R1 增加逻辑的清晰度	R1/R2 取决于使用的语言	R1/R2 取决于使用的语言	——	——
6	有限制的使用递归	——	——	R1/R2 取决于使用的语言	-	R1/R2 取决于使用的语言	R1/R2 取决于使用的语言	——	——
7	在更高级语言程序中无非结构化控制流	——	——	R1/R2 取决于使用的语言	R1 增加逻辑的清晰度	R1/R2 取决于使用的语言	R1/R2 取决于使用的语言	——	——
8	无自动类型转换	——	R2 防止舍入错误	R2 防止舍入错误	R1	R1	——	——	——

表 C.12　详细属性—动态分析和测试

（被表 B.2 参考）

技术/措施		属性			
		针对软件设计规范，所做测试和验证的完整性	针对软件设计规范（成功完成），所做测试和验证的正确性	可重复性	精确定义的测试和验证配置
1	根据边界值分析执行测试用例	——	R1 （如果对边界结果有客观准则，则 R2）	——	——
2	根据错误推测执行测试用例	——	R1	——	——
3	根据错误植入执行测试用例	——	R1	——	——
4	根据基于模型测试用例的生成执行测试用例	R2 MBT过程从要求开始，并有助于早期发现在软件的设计和开发中的错误 R3 如果严格的推理	R2 结果评估和回归测试套件是 MBT 的一个主要获益点，有助于对特定要求结果的理解 R3 如果运用严格的建	R3 MBT（带 TCG）旨在生成测试的自动执行	R2 MBT 是自动的，测试配置要精确定义；生成的测试的执行和黑盒测试类似，这些黑盒测试具有与源代码级覆盖测量结合的可能性

		应用于建模，并且使用了测试用例生成（TCG）	模方法，那么客观的覆盖证据是可能的		
5	性能建模	——	R1 （如果有客观性能要求，则R2）	——	——
6	等价类和输入划分测试	R1 （如果输入数据剖面是良好定义的并且在结构上是易于管理的）	R1 （如果划分在表面上不包含非线性关系，即一个分类里的所有成员是真正等效的）	——	——
7	基于结构的测试	——	R1 （有客观结构覆盖目标，则R2）	——	——

表 C.13　详细属性——功能和黑盒测试

（被表 B.3 引用）

技术/措施		属性			
		针对设计规范，测试、集成和确认的完整性	针对设计规范（成功完成）的测试、集成和确认的正确性	可重复性	精确定义的测试、集成和确认配置
1	根据因果图执行测试用例	R1	R1	-	-
2	根据基于模型测试用例的生成执行测试用例	R2 MBT 依据系统要求和规定的功能的模型自动生成有效的测试用例/规程，有利于早期错误暴露和对规定要求的后果的理解 R3 如果严格的推理应用于建模，并且使用了 TCG	R2 MBT 基于由（主要功能/行为）要求得到的系统模型 R3 如果运用严格的建模方法，那么客观的覆盖证据是可能的	R3 MBT（带 TCG）的目的是自动执行所生成的测试。	R2 MBT 是自动的，测试配置要精确定义
3	原型设计/动画	-	R1	-	-
4	等价类和输入划分测试，包括边界值分析	R1 （如果输入数据剖面是良好定义的并且在结构上是	R1 （如果划分在表面上不包含非线性关系，即一个分	-	-

		易于管理的）	类里的所有成员是真正等效的）		
5	过程仿真	–	R1	–	R2 提供了外部环境的定义

表 C.14　详细属性——失效分析

（被表 B.4 引用）

技术/措施		属性						
		针对本标准，功能安全评估的完整性	针对设计规范（成功完成），功能安全评估的正确性	所有被识别问题的可追溯的闭环	不需要大量的重新评估的变更后，修改功能安全评估的能力	可重复性	及时性	精确定义配置
1a	因果图	R2	R2	–	–	–	–	–
1b	事件树分析	R2	R2	–	–	–	–	–
2	故障树分析	R2	R2	–	–	–	–	–
3	软件功能失效分析	R2	R2	–	–	–	–	–

表 C.15　详细属性——建模

（被表 B.5 引用）

技术/措施		属性			
		针对软件设计规范，确认的完整性	针对软件设计规范（成功完成），确认的正确性	可重复性	精确定义的确认配置
1	数据流图	–	R1	–	–
2a	有限状态机	R3	R3	–	–
2b	形式化方法	R3	R3	–	–
2c	时间佩特里（Petri）网	–	R1	–	–
3	性能建模	–	R1	–	–
4	原型设计/动画	–	R1	–	–
5	结构图	–	R1	–	–

表 C.16　详细属性——性能测试

（被表 B.6 引用）

技术/措施		属性			
		针对设计规范，测试和集成的完整性	针对设计规范（成功实现），测试和集成的正确性	可重复性	精确定义的测试和集成配置
1	雪崩/压力测试	–	R1 （如果设置了客观目标，则 R2）	–	–
2	响应时间和存储约束	–	R1 （如果设置了客观目标，则 R2）	–	–
3	性能要求	–	R1 （如果设置了客观目标，则 R2）	–	–

表 C.17　详细属性——半形式化方法

（被表 B.7 引用）

技术/措施		属性									
		针对软件安全要求规范的完整性	针对软件安全要求规范的正确性	防止固有设计故障	安全要求的可理解性	没有非安全功能对由软件来确保安全的不利干扰	简单性和易懂性	行为可预测性	可验证和可测试的设计	故障裕度/故障检测	防止外部事件造成的共因失效
1	逻辑/功能块图	R2	R2	R2	–	–	R1	R2	–	–	R1
2	时序图	R2	R2	R2	–	–	R1	R2	–	–	R2
3	数据流图	R1	R1	R1	–	–	R1 适用于事务处理	–	–	–	R1
4a	有限状态机/状态转换图	R2	R2	R2	–	–	R1 用数学方法完成了事件顺序的规范	R2	–	–	R2
4b	时间佩特里（Petri）网	R2	R2	R2	–	–	R1 规定实时交互	R2	–	–	R2
5	实体-关系-属	R1	R1	R1	–	–	R1	–	–	–	R1

	性数据模型										
6	消息序列图	R2	R2	R2	–	–	R1	R2	–	–	R2
7	判定/真值表	R2	R2	R2			R1 用于组合逻辑	R2			R2

表 C.18　系统性安全完整性的属性——静态分析

（被表 B.8 引用）

技术/措施		属性			
		针对前一阶段验证的完整性	针对前一阶段（成功完成），验证的正确性	可重复性	精确定义的验证配置
1	边界值分析	–	R1 （如果边界结果有客观准则，则 R2）	–	–
2	检查表	–	R1	–	R1
3	控制流分析	–	R1	–	–
4	数据流分析	–	R1	–	–
5	错误推测	–	R1	–	–
6a	形式化检查，包括具体的准则	R2	R2	–	R2
6b	走查（软件）	R1	R1	–	R1
7	符号化执行	–	R2 如果在正式定义了前提条件和后置条件的场合使用，并使用了一种数学上严格算法的工具实施，则 R3	–	–
8	设计复审	R2	R1 R2（带客观准则）	–	R2
9	运行时错误行为的静态分析	–	R1 对于特定种类的错误，如果使用了一种数学上严格算法的工具实施，则 R3	–	–
10	最坏情况的执行时间分析	R1	R3	–	R2

表 C.19　详细属性——模块化方法

（被表 B.9 引用）

技术/措施		属性							
		针对软件安全要求规范的完整性	针对软件安全要求规范的正确性	防止固有设计故障	简单性和易懂性	行为可预测性	可验证和可测试的设计	故障裕度/故障检测	防止共因失效
1	软件模块规模限制	–	–	R1	R1	R1	R1	–	–
2	软件复杂度控制	–	–	R1	R1	R1	R1	–	–
3	信息隐藏/封装	–	–	R1	R1	R1	R1	–	–
4	参数数量限制/固定数量的子程序参数	–	–	R1	R1	R1	R1	–	–
5	子程序和函数的单入口/单出口	–	–	R1	R1	R1	R1	–	–
6	充分定义的接口	–	–	R2	R1	R1	R1	–	–

附 录 D
（规范性附录）
符合项安全手册—软件组件的附加要求

D.1 安全手册的目的

D.1.1 当一个组件被重复使用或打算被一个或多个其他系统开发重复使用时，有必要确保此组件具有足够的、严格的、完整的说明（即功能、约束和证据），以便可以评估全部或部分依赖于该组件的特定安全功能的完整性。这应依靠安全手册来实现。

D.1.2 如果能够满足GB/T 20438.2附录D和本附录的要求，安全手册可以由组件的供货厂家的文档构成。否则，应作为安全相关系统的设计的一部分建立安全手册。

D.1.3 安全手册应定义组件的属性，其中可能包含集成商在应用时应了解并考虑的硬件约束和/或软件。安全手册是通知集成商该组件属性、设计目的、行为和特性的重要载体。

注1：安全手册的范围和交付时间将取决于其使用人员、集成商类型、组件的目的及提供和维护该组件的人员。

注2：集成商指集成软件的人员、部门或组织。

D.2 软件组件安全手册的内容

D.2.1 安全手册应包含由GB/T 20438.2附录D所要求的与此组件紧密相关的全部信息。例如，GB/T 20438.2附录D中与硬件相关的条款对纯软件组件来说是无关的。

D.2.2 组件应被标识，且集成商可获得所有为正确使用该组件的必要说明。

注：对软件，可通过明确标识组件和证明其内容未被更改来证实。

D.2.3 组件配置：

a) 安全手册应说明软件组件的配置、软件和硬件的运行时环境、以及编译/链接系统的配置（如有必要）。

b) 安全手册应说明软件组件的推荐配置，并且应在安全应用中使用该配置。

c) 安全手册应包括使用该组件依据的所有假设。

D.2.4 安全手册中应包含以下内容：

a) 能力：应规定该组件的集成商所需的最低知识程度，即具体应用工具的知识。

b) 分配给组件的置信度：组件的任何证书的细节、已执行的独立评估、集成商能分配给已有组件的完整性。这应包括组件设计的完整性、在设计阶段所遵循的标准，以及传达给集成商的为了支撑所声称的系统性能力而必须被实施的任何约束。（与组件的功能有关，有可能有些要求仅在系统的集成阶段被满足。在这种情况下，这些要求应由集成商为以后的进展进行识别。与响应时间和性能有关的要求是两个这方面的例子）。

注：与GB/T 20438.2不同，GB/T 20438.3不要求在符合项安全手册中包含软件失效模式或定量的失效率，因为软件错误的原因从根本上不同于GB/T 20438.2附录D关注的随机硬件失效的原因。

c) 安装说明：将已有组件安装到集成系统中的方法的细节或参考。

d) 发布组件的原因：已有组件消除显著的异常或增加附加功能的细节。

e) 显著异常：应给出所有显著异常的细节，包括对异常的解释，异常是如何产生的，以及集成商使用特殊功能减缓异常的机制。

f) 向后兼容：组件是否与之前发布的子系统兼容的细节；如果不兼容，应遵守的升级路径过程的细节。

g) 与其他系统的兼容性：已有的组件可能依赖于一个专门开发的操作系统。此时，应给出该专门开发的操作系统版本的细节。

还应说明构建标准，包含编译器的标识和版本、创建已有组件使用的工具（标识和版本）、以及已有组件使用的测试（标识和版本）。

h) 组件配置：应给出已有组件的名称和描述的细节，包括版本/发行号/修改状态。

i) 变更控制：集成商可向软件生产商发起变更请求的机制。

j) 未达到的要求：可能存在已规定的具体要求在组件的当前版本不能满足的情况。此时，应指明这些要求供集成商考虑。

k) 设计安全状态：在某些情形下，由于系统应用发生受控的失效，组件可能回复到一种设计安全状态。此时，应说明设计安全状态的准确定义供集成商考虑。

l) 接口约束：任何具体约束的细节、特别是用户接口要求必须指明。

m) 任何可能用于防范已列出的威胁和漏洞的安保措施。

n) 可配置的组件：应提供组件的可用配置方法的细节、用途和任何使用约束。

D.3 符合项安全手册中声明的论证

D.3.1 在安全手册中对符合项的所有声明应有足够的支持证据来论证。参见GB/T20438.2的7.4.9.7。

注1：一个组件声称的安全性能有充分的证据支持是必要的。无证据支持的声明无助于建立该组件所提供的安全功能的正确性和完整性。

注2：支持证据可源自组件供应商自身的文档和组件供应商开发过程的记录，或由安全相关系统的开发者或第三方通过附加的评定活动创建或补充。

注3：能否获得证据可能受限于商业或法律约束（例如，版权或知识产权）。这些约束不在本标准的范围之内。

D.3.2 用于论证符合项安全手册中声明的证据与组件安全手册是不同的。

D.3.3 对于无法为功能安全评估提供证据的情形，则该组件不适合用于E/E/PE安全相关系统。

附 录 E
（资料性附录）
GB/T 20438.2 和 GB/T 20438.3 之间的关系

下表有助于寻找GB/T 20438.2的哪些条款仅是软件人员需要考虑的，而哪些条款是可以忽略的。众所周知，绝大多数条款是有关硬件的，因此这里不再重复。重要的软件描述由GB/T 20438.3做出，然而，许多软件相关的要求也出现在GB/T 20438.2中，大多与GB/T 20438.3的要求重叠。对那些考虑硬件和软件兼容性的软件专家，需考虑GB/T 20438.2的知识。GB/T20438.2的要求被分为以下类别：

表 E.1 GB/T 20438.2 要求分类

软件	针对关注硬件的和关注软件的本标准的用户
应用软件	针对关注解决相关的安全功能及类似功能的软件，而不是操作系统软件或库函数的用户
系统软件	针对主要关注操作系统软件和库函数及类似内容的用户
仅硬件	针对对软件不感兴趣的用户
主要是硬件	针对仅少量关注软件的用户

表 E.2 GB/T 20438.2 的软件相关要求及其与特定类型软件的典型关联

GB/T 20438.2 要求	用户关注的重点	备注
7.2	软件	
7.2.3.1	应用软件	
7.2.3.2 到 7.2.3.6	软件	
7.2.3.3	仅硬件	
7.3	软件	7.3.3.2 f）仅硬件
7.4	软件	
7.4.2.1 到 7.4.2.12	软件	
7.4.2.13、7.4.2.14	仅硬件	
7.4.3.1 到 7.4.3.3	软件	
7.4.3.4	仅硬件	
7.4.4	仅硬件	
7.4.5	仅硬件	
7.4.6	软件	7.4.6.7 仅硬件
7.4.7	软件	7.4.7.1 a), b) 仅硬件
7.4.8	仅硬件	
7.4.9.1 到 7.4.9.3	软件	

GB/T 20438.2 要求	用户关注的重点	备注
7.4.9.4、7.4.9.5	仅硬件	
7.4.9.6、7.4.9.7	软件	
7.4.10	软件	主要为系统软件
7.4.11	仅硬件	
7.5	软件	
7.6	软件	
7.6.2.1 a)	仅硬件	
7.6.2.4	主要是硬件	
7.7	软件	7.7.2.3，7.7.2.4 主要是应用软件
7.8	软件	
7.9	主要是应用软件	
8	软件	
附录 A.1	主要是硬件	
附录 A.2 及表	主要是硬件	表 A.10 软件
附录 A.3	主要是硬件	表 A.16，A.17，A.18 包含若干软件方面
附录 B、所有表	软件	
附录 C	仅硬件	
附录 D	软件	D.2.3 仅硬件
附录 E	仅硬件	
附录 F	仅硬件	

附 录 F
（资料性附录）
单计算机中各软件组件间实现互不干扰的技术

F.1 概述

单计算机系统（由一个或多个处理器和这些处理器间共享的内存和其他硬件设备组成）中运行的各软件组件间的执行的独立性可通过多种方法实现和证明。本附录列出了可用于实现互不干扰（不同系统性能力的组件之间，或为实现同一安全功能或对同一安全功能有贡献的组件之间，或对某一安全功能有贡献的软件和同一计算机中的非安全相关软件之间）的若干技术。

注：术语“执行的独立性”指各组件的执行行为之间没有可能产生危险失效的相互负面干扰。它用于区分在组件间要求的其他方面的独立性，尤其是，为了满足本标准其他要求的多样性。

F.2 行为域

执行的独立性应同时在空间和时间域上实现和证明。

空间域：某组件使用的数据必须不能被另一组件更改。尤其是，不应被某非安全相关组件更改。

时间域：某组件不应由于占用过多可用的处理器执行时间份额、或通过锁定某种资源以阻止其他组件的执行而导致其他组件无法正确执行功能。

F.3 原因分析

为证明执行的独立性，应对提出的设计进行分析以辨识在空间和时间域名义上独立的（互不干扰）各组件间执行干扰的所有可能原因。分析应既考虑正常运行也考虑失效情况下的运行，并且应包括（但不限于）以下内容：

a) 共享随机存储器；

b) 共享外设；

c) 共享处理器时间（当两个或多个组件由单个处理器执行时）；

d) 为实现总体设计所必需的各组件间的通讯；

e) 某组件的失效（如溢出、或除零异常、或错误的指针运算）可能导致其他组件继发失效的可能性；

执行的独立性的实现和论证应考虑所有以上辨识出的干扰来源。

F.4 实现空间域独立性

用于实现和证明空间域独立性的技术如下：

a) 对不同组件，包括对不同系统性能力的组件，使用硬件存储保护。

b) 使用允许各组件在其自有虚拟存储空间内执行其自有进程的操作系统，并有硬件存储保护支持。

c) 使用严格的设计、源代码和可能的目标代码分析以证明各组件间没有可能导致属于其他组件的数据被改写的显性或隐性的内存引用（适用于硬件存储保护不可用的情形）。

d) 软件保护高完整性组件的数据不会被低完整性组件非法更改。

除非高完整性组件可验证数据具有足够的完整性，否则数据不应从低完整性组件传送给高完整性组件。

当数据必须在要求独立的组件间传递时，应优先使用单向接口，如消息或管道，而尽量避免使用共享内存。

注：理想情况下独立的组件相互间应不会通讯。但当系统设计要求一个组件应向另一个组件发送数据时，通讯机制应设计为，如果数据传输停止或延迟时，无论发送组件还是接收组件都不应失效或被阻塞执行。

除随机存储器中的瞬时数据外，任何存储在如磁盘等永久性存储设备上的数据都必须考虑空间划分。例如，可通过操作系统实现的文件访问保护以防止一个组件向属于另一组件的数据区域写入数据。

F.5 实现时间域独立性

用于保证时间域独立性的技术如下：

a) 确定性调度方法。例如：

——一种向每一组件提供一个固定的时间片并辅以各组件的最坏情况执行时间分析的周期性调度算法，以静态地证明各组件满足时间要求。

——时间触发架构。

b) 基于严格优先级的调度，由一个实时执行器及一种避免优先级反转的方法实现；

c) 时界，用于当某组件运行超过分配的执行时间或截止时间时即终止其执行（这种情况下，必须实施危险分析以证明该组件的终止不会导致危险失效，因此该技术可能最好在非安全相关组件中采用）；

d) 一种操作系统，可保证没有进程会无法获取处理器时间，例如采用时间片方式。此方法仅用于满足没有硬实时要求的安全相关组件，并需表明此调度算法不会对任何组件产生不当的延迟。

当各个组件共享一个资源时（如一个外设），设计应保证组件不会因共享资源被其它组件锁住，而不能正常执行功能。在确定时间域的互不干扰性时，应考虑访问共享资源要求的时间。

F.6 对支持软件的要求

如果使用操作系统、实时执行器、存储管理、定时器管理或任何其他类似软件提供空间域或时间域独立性或二者均提供，则这种软件必须在所有要求独立性的组件中具有最高的系统性能力。

注：显然，任何这种软件都代表了独立组件的一个潜在失效的共同原因。

F.7 软件模块独立性——编程语言方面

下表F.1是一个有关术语的非正式定义。

表 F.1　模块耦合——术语定义

术语	非正式定义
内聚性	一个模块内部的数据和子程序间联系的紧密性度量
耦合度	模块间联系的紧密性度量
封装	将内部（私有）数据和子程序对外界访问隐藏；该术语主要用于面向对象的程序
独立性	软件部分的解耦度量；与耦合度互补
模块	执行某些功能且可能有自有数据的封闭软件部分；依照编程语言，例如 class（类）、类的层次结构、子程序、unit（单元）、module（模块）、package（包）……
接口	供访问模块的明确定义的子程序头的集合
透传数据（Tramp data）	在接收模块中不使用、仅被传递到其他模块的数据

作为普遍的规则，如果模块间耦合度低并且模块内聚性高，则模块的独立性增强。高内聚性促进功能的可识别单元和代码实现的可识别单元能明确对应，低模块耦合度能减少交互从而提升功能不相关的模块间的高独立性。

低模块耦合度通常源自将用于执行某特殊功能的代码和数据放在一起而实现的模块的高内聚性。如果代码和数据在模块间随意放置，或受限于某些时序或控制流顺序的原因，则会导致低内聚性。

模块耦合可区分为若干方面，见下方表F.2。

表 F.2　模块耦合类型

耦合	定义	解释	理由	备注
接口耦合，封装	仅通过明确定义的一组子程序耦合。	仅通过子程序访问模块或其数据；一个变量的一个值的任何改变、或与该变量值有关的任何问题、或该模块被要求的任何其他服务都通过子程序调用的途径实现。	一个模块的子程序的头（签名）解释了提供的服务。 如果模块需要任何变更，模块内可做大量的变更而不影响其他模块。 此方法可降低耦合度，通常推荐。	主要适用于面向对象程序、类、类的层次结构、库的包，而不适用于子程序。
通过参数表的数据耦合	数据仅通过参数表或子程序的标识符传送。	仅通过在子程序头中指明的变量或对象访问模块或其数据；一个变量的一个值的任何改变、与该变量的值有关的任何问题都是可见的。	子程序的头列出了与调用该子程序有关的数据或对象。 此方法可降低耦合度，通常推荐。	在面向对象程序的类中一般不遵守该原则。可能直接访问局部变量。严格遵守该原则可能会产生透传数据。为避免这类数据应不遵守该原则。

结构耦合	数据传输包含多余的数据。	为执行要求的功能向接收子程序传输了多余的数据。	多余的数据向另一模块提供了不要求满足其目的的信息。这些数据可能导致模块间的协作产生误解。但并不反对使用这种做法。	该缺陷通常易于纠正。
控制耦合	在接收模块上实施立即控制的耦合。	数据传输只能导致另一模块的分支反应；很多情形下通过传输单个比特的形式出现。	比上述耦合更紧密，因为它要求立即动作，命令接收子程序做某些操作。须谨慎对待；如果可能应避免。通常不推荐。	一般难以避免。可能是必要的，例如通知某个行为的完成，或一个数据的有效性。
全局耦合	通过全局数据耦合。	模块可访问其他模块能直接访问的数据，或一个模块可直接访问属于另一个模块的数据。	子程序的头未指明使用的数据及来源。难以理解子程序的功能和预测代码变更后的影响。	通常反对使用。在个别情况下可能有必要，例如为避免透传数据。只在遵守明确定义和记录的编码规范的有限情形下使用。
内容耦合	直接跳转入其他模块，影响其他模块的分支目标，或直接访问其他模块的数据。	在汇编语言程序中可行；在所有高级语言中都不可行；能加速程序的执行和编码工作。	反对使用。只能通过理解其关联模块来理解一个模块。使程序极其难以理解和更改。	在某些编程语言中甚至不可能。可始终避免。

代码阅读或代码复审（见7.9.2.12）应验证程序模块是否为低耦合的。该分析通常要求对模块目的和工作方式的某种理解。因此，只能通过阅读代码及其文档来评估合适的耦合度。

内容耦合应避免。全局耦合仅在极个别情况下使用。控制耦合和结构耦合应避免。只要可能，模块间应通过接口耦合（封装）和/或数据耦合相联系。

附 录 G
（资料性附录）
数据驱动系统的生命周期裁剪指南

G.1 数据驱动——系统部分和应用部分

很多系统由两部分写成。一部分提供基础的系统能力。另一部分使系统适用于预期应用的特定要求。应用部分可能以数据的形式写成，用于配置系统部分。此即本附录中的术语“数据驱动”。

软件针对于应用的部分可能通过各种编程工具和编程语言开发。这些语言和工具可能约束了编写应用程序的方式。

例如，当一种编程语言支持开发者/配置者描述功能（例如，对简单联锁系统使用梯形逻辑），则应用软件编程的任务会相对简单。然而，当编程语言允许开发者/配置者描述复杂应用行为，则应用软件编程的任务会相对复杂。当开发了十分简单的应用软件时，详细设计可能被考虑为配置而不是编程。

为实现要求的安全完整性所必需的严格度，取决于提供给开发者/配置者的配置复杂程度和应用所表达的行为复杂程度。这图形化地体现在图G.1的坐标轴中。

为简单起见坐标轴进一步分为以下复杂度类别：

a) 语言允许的可变性：
   ——固定程序；
   ——有限可变性（某些工业把应用程序视为被系统部分解析的“数据”）；
   ——全可变性（尽管通常不被考虑为数据驱动，这类系统仍在应用开发中使用，并为完整起见包括在本附录中）。
b) 配置应用的能力:
   ——有限的；
   ——完全的。

现实中一个特定的系统可由不同级别的复杂度和可配置性组成。更进一步，复杂度可沿两个坐标轴的连续区间表现出可变的规模。当试图裁剪软件生命周期时，应辨识有关的复杂度等级，并应论证裁剪的程度。

下面给出了对应每一复杂度等级的典型系统类型的描述。实现每种系统的建议的技术指南在GB/T 20438.7中给出。

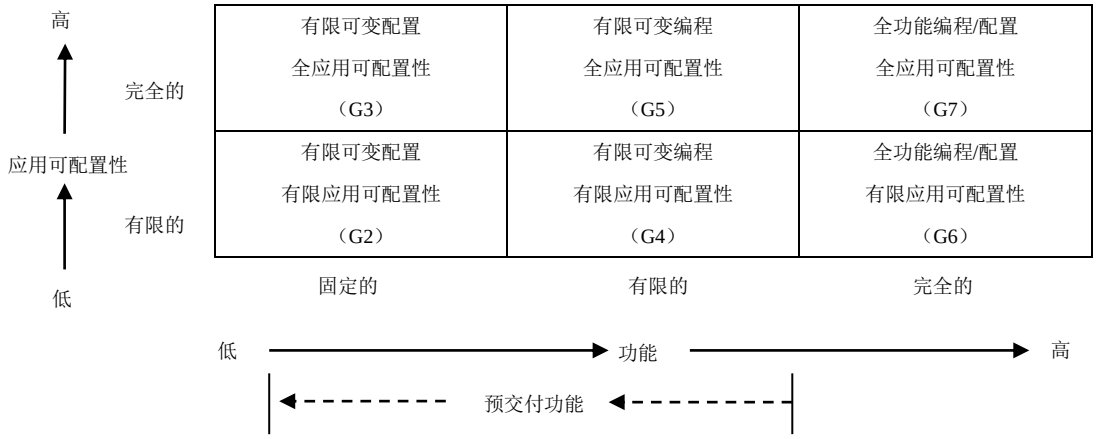

图 G.1　数据驱动系统的复杂度中的可变性

每种复杂度类别的典型系统在G.2中描述。

G.2　有限可变配置、有限应用可配置性

一种专有的配置语言与一种带固定预交付功能并符合GB/T 20438的系统共同使用。

配置语言不允许编程者改变系统的功能，而是将配置限制在通过调整少量（数据）参数使系统与应用相匹配。例如智能传感器和执行器（可输入特定参数）、网络控制器、顺序控制器、小型数据记录系统和智能仪表。

裁剪安全生命周期的论证应包括但不限于以下内容：

a)　关于该应用的输入参数的规范；

b)　验证在操作系统中正确实施了参数；

c)　输入数据的所有组合的确认；

d)　配置过程中特定的和具体操作模式的考虑；

e)　人为因素/人体工程学；

f)　联锁，例如保证正在运行的联锁在配置过程中不会失去有效性；

g)　无意的重复配置，例如钥匙开关访问、保护设备。

G.3　有限可变配置、全应用可配置性

一种专有的配置语言与一种带固定预交付功能并符合GB/T 20438的系统共同使用。

配置语言不允许编程者改变系统的功能，而是将配置限制在通过创建大量的静态数据参数使系统与应用相匹配。例如一个空中交通管制系统，包含大量数据实体，每个数据实体带有一个或多个属性。数据的重要特性是数据中不包含明确的顺序、次序或分支结构，并且不包含任何形式应用的组合状态。

除了在G.2给出的考虑外，裁剪安全生命周期的论证应包括但不限于以下内容：

a)　创建数据的自动化工具；

b)　一致性检查，例如数据是自兼容的；

c)　规则检查，例如确保数据的生成符合规定的约束；

d)　与数据准备系统的接口的有效性；

G.4 有限可变编程、有限应用可配置性

一种问题导向的语言与一种符合GB/T 20438的系统共同使用，其中语言的语句包含或类似于用户应用的术语，该术语针对具有有限预交付功能的系统。

这些语言允许用户根据自身特定的要求，基于一系列硬件和软件组件定制系统功能的有限的灵活性。

有限可变编程的一个基本特点是数据可能包含明确的顺序、次序或分支结构，并且可能调用应用的组合状态。例如功能块编程、梯形逻辑、表格化系统及图形化系统。

除在G.3中给出的考虑之外，应包括但不限于以下内容：

a) 应用要求的规范；
b) 针对该应用所允许的语言子集；
c) 组合语言子集的设计方法；
d) 针对潜在的系统状态组合的验证的覆盖准则。

G.5 有限可变编程、全应用可配置性

一种问题导向的语言与一种符合GB/T 20438的系统共同使用，其中语言的语句包含或类似于用户应用的术语，该术语针对具有有限预交付功能的系统。

与有限可变编程、有限应用可配置性的基本区别在于应用配置的复杂程度。例子如图形化系统和基于SCADA的批量处理控制系统。

除在G.4中给出的考虑之外，应包括但不限于以下内容：

a) 应用的架构设计；
b) 模板的规定；
c) 单个模板的验证；
d) 应用的验证与确认。

本标准中概括的生命周期方面内最底层的模块实施和测试，很可能是不必要的（取决于所用的编程语言）。

G.6 全功能编程/配置、有限应用可配置性

见以下G.7。

G.7 全功能编程/配置、全应用可配置性

对这些系统，本标准的全部生命周期要求都适用。

系统的全可变部分基于通用编程语言或通用数据库语言或常规的科学和仿真程序包。这些部分常与基于计算机的系统结合使用，配备有提供系统资源分配和实时多编程环境的操作系统。可能由全可变语言写成的系统的例子如：专用的机械控制系统、专门开发的飞行控制系统、以及用于管理安全相关服务的网络服务。

参 考 文 献

[1] GB/T 21109-2007(所有部分) 过程工业领域安全仪表系统的功能安全；

[2] GB 28526-2012 机械电气安全 安全相关电气、电子和可编程电子控制系统的功能安全

[3] IEC 61800-5-2 Adjustable speed electrical power drive systems - Part 5-2: Safety requirements - Functional

[4] IEC 61508-5:2010 Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 5: Examples of methods for the determination of safety integrity levels

[5] IEC 61508-6:2010 Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

[6] IEC 61508-7:2010 Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 7: Overview of techniques and measures

[7] GB 9706 (所有部分) 医用电气设备

[8] GB/T 15969.3-2005 可编程序控制器 第3部分:编程语言