ICS 11.040.01
C 30

YY

中华人民共和国医药行业标准

YY/T 1406.1—2016/IEC/TR 80002-1:2009

医疗器械软件 第1部分:YY/T 0316应用于医疗器械软件的指南

Medical device software—Part 1:Guidance on the application of ISO 14971 to medical device software

(IEC/TR 80002-1:2009,IDT)

2016-03-23 发布　　2017-01-01 实施

国家食品药品监督管理总局　发布

中华人民共和国医药
行业标准
医疗器械软件
第1部分：YY/T 0316 应用于医疗器械软件的指南
YY/T 1406.1—2016/IEC/TR 80002-1:2009

*

中国标准出版社出版发行
北京市朝阳区和平里西街甲2号(100029)
北京市西城区三里河北街16号(100045)

网址 www.spc.net.cn
总编室：(010)68533533 发行中心：(010)51780238
读者服务部：(010)68523946

中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销

*

开本 880×1230 1/16 印张 3.5 字数 104 千字
2017年2月第一版 2017年2月第一次印刷

*

书号：155066·2-31114 定价 48.00 元

目　次

前　言

YY/T 1406《医疗器械软件》，由下列部分组成：

——第1部分：YY/T 0316 应用于医疗器械软件的指南；

——第2部分：医疗器械质量体系软件的确认；

——第3部分：医疗器械软件生存周期过程（YY/T 0664）的过程参考模型。

本部分为 YY/T 1406 的第1部分。

本部分按照 GB/T 1.1—2009 给出的规则起草。

本部分使用翻译法等同采用 IEC/TR 80002-1:2009《医疗器械软件　第1部分：ISO 14971 应用于医疗器械软件的指南》（英文版）。

本部分方框中的文本内容直接引用自 YY/T 0316—2016 标准，在文本的前面写明“YY/T 0316—2016 原文”。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本部分由国家食品药品监督管理总局提出。

本部分由国家食品药品监督管理总局医疗器械标准管理中心归口。

本部分起草单位：北京国医械华光认证有限公司、上海市医疗器械检测所、沈阳东软医疗系统有限公司。

本部分主要起草人：王志强、由洪顺、米兰英、何骏、陆锷、郑一菡、陈志刚。

引　言

软件通常作为医疗器械技术的不可或缺的组成部分。对于包含软件的医疗器械建立其安全和有效性，需要知道软件的预期用途，同时要证明软件的实现满足预期用途而不引起任何不可接受的风险。

虽然软件本身不是一种危险(源)，但软件可能引发危险情况，理解这一点很重要。宜总是以系统观点来考虑软件，软件的风险管理不能脱离系统孤立地进行。

复杂的软件设计可能涉及复杂的事件序列，这些序列可能引发危险情况。软件风险管理的任务包含识别那些导致危险情况的事件序列，识别在这些事件序列中哪些位置可以中断序列，预防伤害或降低伤害概率。

引发危险情况的软件事件序列可分为两类：

a) 事件序列表现为软件对输入产生不可预见的响应(软件规范中的错误)；

b) 事件序列是由编码错误引起的(软件实施中的错误)。

由于正确地规范和实施复杂系统的难度以及完整验证复杂系统的难度，所以这些分类对软件来说是特有的。

因为很难估计会引发危险情况的软件异常的概率，并且因为软件在使用中不会因为损耗而随机失效，所以软件方面的风险分析，宜关注潜在软件功能的识别和会导致危险情况的异常的识别——而不是估计概率。软件异常引发的风险大多数情况下只需要评价伤害的严重度。

风险管理通常是有挑战性的，当包含软件时更是如此。下面的条包含了关于软件特性的补充细节，这为从软件层面理解 YY/T 0316—2016 提供了指南。

● **本部分的结构：**

本部分遵循了 YY/T 0316—2016 的结构，并且为与软件相关的每项风险管理活动提供了指南。

由于软件生命周期中风险管理活动的迭代特性，在提供的信息中存在一些有意的冗余。

医疗器械软件
第1部分:YY/T 0316应用于医疗器械软件的指南

1 总则

1.1 范围

本部分为YY/T 0316—2016《医疗器械 风险管理对医疗器械的应用》中包含的要求应用于有关YY/T 0664—2008《医疗器械软件 软件生存周期过程》中所指的医疗器械软件提供了指南,本部分并不增加或改变YY/T 0316—2016或YY/T 0664—2008的要求。

当软件作为医疗器械/系统时,本部分供需要实施风险管理的风险管理从业者以及需要理解如何满足YY/T 0316—2016中阐述的风险管理要求的软件工程师使用。

监管机构意识到ISO 14971在世界范围内被广泛认可作为实施医疗器械风险管理的重要标准。YY/T 0664—2008规范性引用了YY/T 0316—2016中的要求。这两个标准为本部分提供了基础。

宜说明的是,虽然YY/T 0316—2016和本部分关注的是医疗器械,但本部分还可用于为医疗卫生保健环境中的所有软件实施安全风险管理过程,无论该软件是否被归类为医疗器械。

本部分不涉及:

——已经由现有标准或计划中的标准覆盖的领域,如:报警、可用性工程、网络等;

——生产或质量管理体系软件;

——软件开发工具。

本部分预期不作为法规检查或认证评定活动的依据。

本部分中"宜"是用来表明,在满足要求的若干可能性中,推荐特别适合的一种,并未提及或排斥其他的可能性,或者用来表明某种做法更好但不是必须的要求。"宜"不应理解为要求。

1.2 规范性引用文件

下列文件对于本部分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)适用于本部分。

YY/T 0316—2016 医疗器械 风险管理对医疗器械的应用(ISO 14971:2007更正版,IDT)

YY/T 0664—2008 医疗器械软件 软件生存周期过程(IEC 62304:2006,IDT)

2 术语和定义

YY/T 0316—2016和YY/T 0664—2008界定的以及下列术语和定义适用于本文件。

注:定义术语的索引开始于49页。

2.1

多样性 diversity

一种冗余的形式,冗余要素用于不同的(多样的)组件、技术或方法以降低共同原因导致所有要素同时失效的概率。

2.2

冗余 redundancy

提供多个组件或机制来完成同样的功能，使得一个或多个组件或机制的失效不会妨碍功能的执行。

2.3

安全相关软件 safety-related software

能够引发危险情况的软件，或用于实现风险控制措施的软件。

3 风险管理通用要求

3.1 风险管理过程

3.1.1 总则

YY/T 0316—2016 原文

3 风险管理通用要求

3.1 风险管理过程

制造商宜在整个生命周期内，建立、形成文件和保持一个持续的过程，用以识别与医疗器械有关的危险(源)，估计和评价相关的风险，控制这些风险并监视上述控制的有效性。此过程应包括下列要素：

——风险分析；

——风险评价；

——风险控制；

——生产和生产后的信息。

在有形成文件的产品实现过程时，如 YY/T 0287—2003 第 7 章所描述的过程，则该过程应包括风险管理过程中的适当部分。

注 1：形成文件的质量管理体系过程可用于系统地处理安全问题，特别是能够在复杂医疗器械和系统中，对危险(源)和危险情况进行早期判断。

注 2：风险管理过程的示意图见图 1。按照特定的生命周期阶段，风险管理的每个要素可有不同的侧重点。此外，对于某个医疗器械风险管理活动可适当地重复执行或在多个步骤中执行。附录 B 包括了风险管理过程中各个步骤更详细的概述。

用查看适当文件的方法检查符合性。

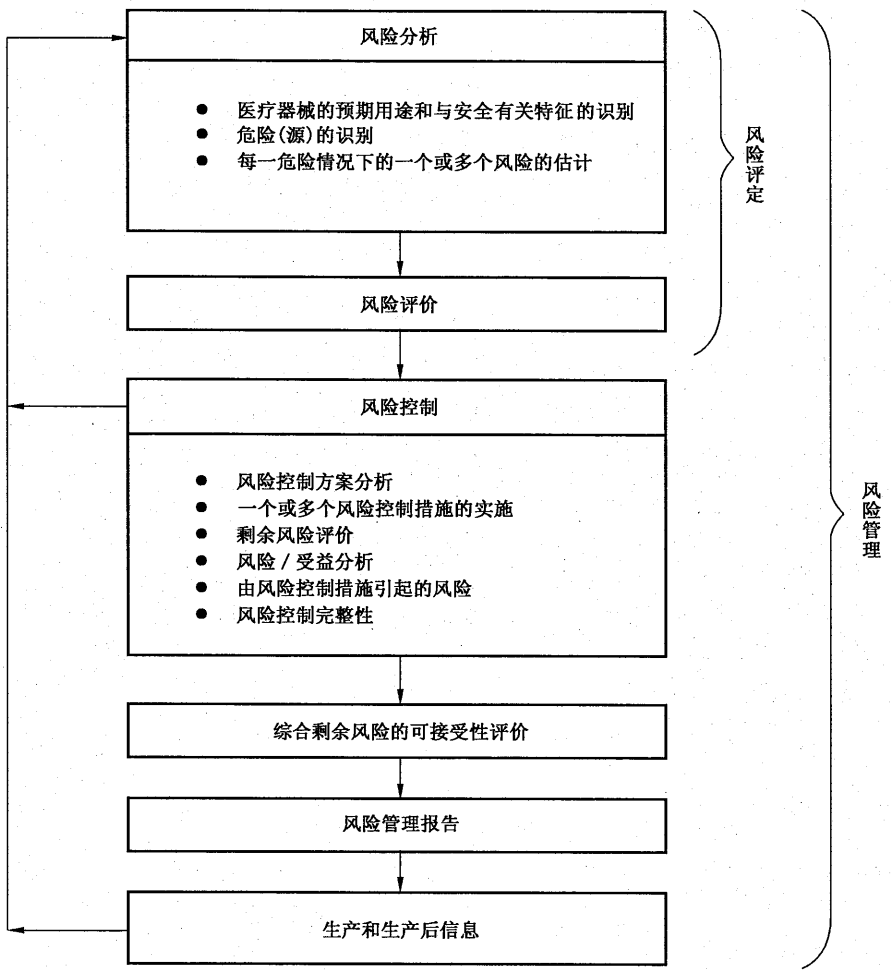


图 1　风险管理过程示意图

安全是系统(这里指整个医疗器械)的一个属性,该系统可能包括软件。风险管理宜在包含软件及其全部硬件环境的系统中进行。软件风险管理活动不宜脱离系统进行。

虽然软件方面的风险管理脱离整体医疗器械风险管理不能有效地执行,但有一些活动作为软件生命周期的不可或缺的组成部分可以由软件工程师很好地完成。与用于整体医疗器械风险管理(YY/T 0316—2016)的要素相比,软件中的有些要素要求更多的关注并有不同的解释。为了做到有效,需要强调的是软件风险管理也需要关注医疗器械风险。

注 1:因为硬件失效、软件失效以及硬件和软件的风险控制措施的互相依赖,如果脱离整体医疗器械的风险管理,软件方面的风险管理无法有效地进行。

注 2:例如,所有的软件失效都是系统性的而非随机的(许多硬件失效/故障是随机的),且其失效的概率无法精确估计。因此,风险的概率要素应用于软件的方式是非常不同的,见 4.4.3。

在医疗器械设计的早期阶段,软件工程师有很多机会致力于医疗器械的总体安全。宜在系统设计定案之前考虑软件在医疗器械安全中的作用。

通过参与医疗器械设计过程,随着设计进展,软件工程师能促使就软件风险做出与安全有关的决策。这些决策宜包括但不限于:

- 提供足够的硬件资源以支持软件;

- 硬件和软件间的功能划分；
- 整体医疗器械的预期用途和软件用户界面的预期用途；
- 避免不必要的复杂软件。

3.1.2 迭代

典型的软件开发生命周期经常使用迭代。使用迭代允许：

- 研究不同软件设计的可行性；
- 在不同时期开发不同的软件项；
- 不同软件版本的阶段性交付；
- 纠正软件开发过程中产生的错误。

在整个软件生命周期中，YY/T 0664—2008 要求风险管理活动是迭代的，并与系统设计活动相协调。例如，在软件开发过程中，当软件需求确定后，YY/T 0664—2008 的 5.2.4 要求对医疗器械风险评定进行重新评价。这种重新评价可导致更新系统需求规范和医疗器械风险评定。风险评价宜在所有阶段重复进行，所有阶段是指从需求经由体系结构和设计直到软件的实现。

YY/T 0316—2016 没有规定设计和开发过程，其通常只要求在设计(包括风险控制措施)实施之前和之后(而不是进行中)完成风险管理步骤。例如，当执行一个风险控制措施时，YY/T 0316—2016 只是要求对这些措施进行评审以确保其不会引起进一步的危险(源)和危险情况。这不宜理解为只是在风险控制措施已经完成后才进行评审，比较有利的做法是：一旦进一步的危险(源)显现出来就进行处置。这意味着在风险控制措施执行中进行迭代。

所有的交付物始终保持一致是很重要的。迭代对交付物的一致性是一种威胁。因此，应用严格的配置管理以保证更改的所有影响都被识别且所有相关的交付物在更改后得到更新是很重要的。当涉及到软件时，这一点尤其重要，因为软件可能很快被更改，看上去小的更改可能产生非预期的负面影响。所有与软件相关的信息都要是最新的，以避免工程师间的错误交流。宜检查软件更改提案的负面影响，特别是影响安全的负面影响。这可能导致风险管理过程的某些部分重复进行。

3.1.3 主动的或被动的安全设计方法

风险管理宜随着医疗器械规范的实质性输入尽早开始，并在设计早期阶段考虑安全，也就是说，主动的设计方法比被动的设计方法更好。在主动方法中，安全与其他客户需求一起被考虑，并转化为早期的安全需求。尽管被动方法有时不可避免(例如当已有的产品升级时)，但主动方法通常是获得安全的医疗器械最有效、最快、最经济的方法。

主动的安全设计的优点是：

——从一开始系统规范不仅包含医疗器械要做什么，还识别了为降低风险宜避免的系统行为；

——从一开始就可以策划一个系统的体系结构，可以证明其提供了想要的特性，同时避免或防止不安全状况；

——因为体系结构已被精心细致地融入全部的设计中，可以在避免重复工作的同时开展风险控制措施；

——安全方法和风险控制措施的选择可以在早期决定(例如，设计的固有安全可被最大化，安全信息可被最小化)。

3.1.4 包含软件的安全的系统的特征

安全的系统高度期望的特征包括：

——使用简单的硬件安全机制以避免对安全相关软件项的过度需求；

——仅使用非常简单的安全相关软件项；

——安全相关软件项分布在若干独立的处理器中；

——需要时有足够的硬件资源，以便能运行所有安全相关软件，避免资源争夺；

——软件时序采用确定性设计。

——适当的处理失效情形，例如：

- 警告用户有失效并且给出知情干预的机会；
- 在失效的情况下提供简化功能；
- 在失效情况下，可能时，安全地关断；
- 从失效中快速恢复。

——具有防止软件代码在其运行环境中通过自行修改或作为数据输入的结果被修改的方法；

——具有检出和/或防止与安全相关的数据损坏的方法。

3.2 管理职责

YY/T 0316—2016 原文

3.2 管理职责

最高管理者应在下列方面对风险管理过程的承诺提供证据：

——确保提供充分的资源；

——确保给风险管理分配有资格的人员(见 3.3)。

最高管理者应：

——规定一个确定风险可接受性准则的方针并形成文件，此方针应确保准则是基于适用的国家或地区法规和相关的标准，并考虑可用的信息，例如通常可接受的最新技术水平和已知的利益相关方的关注点。

——按照计划的时间间隔评审风险管理过程的适宜性，以确保风险管理过程的持续有效性，并且，将任何决定和采取的活动形成文件。如果制造商具有适当的质量管理体系，这些评审可作为质量管理体系评审的一部分。

注：文件可整合进制造商质量管理体系产生的文件中，且这些文件可在风险管理文档中引用。

用查看适当文件的方法检查符合性。

YY/T 0316—2016 和 YY/T 0664—2008 都设定了质量管理体系的恰当的地位。YY/T 0316—2016 的 3.2 列举了风险管理对最高管理者的要求。

注：YY/T 0316—2016 的 3.1 规定风险管理可以是质量管理体系的一个不可或缺的组成部分，并且 YY/T 0664—2008 的 4.1 规定：可以使用符合 YY/T 0287 的质量管理体系或国家法规要求的质量管理体系来证明制造商有能力始终如一地满足客户需求和适用的法规要求。YY/T 0664—2008 的附录 B.4 中也提供了关于 4.1 的规定的指南，说明作为应用适当软件工程方法和技术的整体框架，建立作为质量管理体系的一个不可或缺的组成部分的风险管理是有必要的。

为实现有效的风险管理过程以及对医疗器械软件的安全设计和维护，最高管理者负责将必要的组织结构、充分的资源、职责和培训(见 3.3)落实到位。

制造商可以考虑将软件开发或维护过程活动外包(例如设计、实现、测试或维护)。在这些情况下，最高管理者仍然对确保外包的软件开发或维护过程活动进行适当的风险管理活动，以及对确保风险控制措施的适当应用负全部责任。

当软件开发外包时，制造商宜通过适当的合同来确保对软件及其设计的充分控制，以确保 YY/T 0316—2016 中要求的全部风险管理在医疗器械整个生命周期中都被执行，包括软件发布后软件异常的纠正。

制造商宜考虑对供方设定绩效要求（供方控制见 ISO 13485[1]的 7.4），比如要求供方证实：

——符合 YY/T 0316—2016 的有效的风险管理；

——符合 YY/T 0664—2008 的有效的软件工程实践；

——提供持续满足顾客要求和适用的法规要求的医疗器械软件的能力。

如果对外包的过程或产品应用风险控制措施，风险控制措施及其重要性宜形成文件并在合同中明确的传达给供方。

3.3 人员资格

3.3.1 总则

YY/T 0316—2016 原文

3.3 人员资格

执行风险管理任务的人员，应具有与赋予他们的任务相适应的知识和经验。适当时，应包括特定医疗器械（或类似医疗器械）及其使用的知识和经验、有关的技术或风险管理技术。应保持适当的资格鉴定记录。

注：风险管理任务可以由几种职能的代表执行，每个代表贡献其专业的知识。

用查看适当记录的方法检查符合性。

参与软件系统开发和维护的小组成员宜具有与其承担的任务相适应的知识和经验。对承担风险管理相关任务的人员的基本要求是具有必需的风险管理知识。包括临床专家（诸如临床支持和技术服务专家以及其他相关学科的专家）、软件工程师、系统设计师、可用性/人因工程专家和领域专家的多学科团队的参与，以及他们与软件工程人员和测试人员互动的程度和类型也宜从风险管理角度考虑。

这可能要求为每一个人制定培训计划以确保他们完全理解所要求的活动。

同样，也宜考虑风险管理小组成员在软件方面的资格而且可能要求专门的培训。

以下条提出了宜被考虑的所需知识领域的综述。

3.3.2 预期用途/领域知识

在医疗器械设计的所有阶段，展开对预期用途的认知很重要。这对于软件设计者和软件风险管理人员尤其重要。软件的复杂行为很容易造成用户误用或混淆，导致以前未预见的危险（源）和危险情况。对临床实践进行全面地了解能使风险管理者识别危险（源）和危险情况，使软件工程师避免危险（源）和危险情况，或使软件工程师设计出风险控制措施。

制造商宜确保临床专家（诸如临床支持和技术服务专家及其他相关学科专家）能参与设计活动和风险管理活动，或至少对这些活动提供建议。

另外，制造商宜考虑对软件工程师和风险管理者提供医疗器械临床应用方面的培训。

3.3.3 编程经验和意向

有经验的软件开发人员和测试人员能逐渐认识到在测试中发现所有软件缺陷的难度，因此也能认识到测试后还存在一定比例的软件缺陷。在软件开发团队中包含有经验的人员并且给予其适当的权限来指导、监督和质疑经验少的人员是很重要的。

以下的软件任务分配给有经验的人员尤其重要：

——识别软件失效的方式；

——分析与软件失效有关的风险；

——识别风险控制措施；

——分析软件发布后的问题报告；

——更改的设计和实施，特别是软件发布以后。

在所有这些任务中，通过经验可以意识到软件及软件开发过程出现的错误，并且认识到进行更改的同时保持软件设计完整性的难点。

3.4 风险管理计划

3.4.1 总则

YY/T 0316—2016 原文

3.4 风险管理计划

应策划风险管理活动。因此，对于所考虑特定的医疗器械，制造商应按照风险管理过程，建立一项风险管理计划并形成文件。风险管理计划应是风险管理文档的一部分。

此项计划至少应包括：

a) 策划的风险管理活动范围，识别和描述医疗器械和适用于计划每个要素的生命周期阶段；

b) 职责和权限的分配；

c) 风险管理活动的评审要求；

d) 基于制造商确定可接受风险方针的风险可接受性准则，包括在伤害发生概率不能估计时的可接受风险的准则；

e) 验证活动；

f) 关于相关的生产和生产后信息的收集和评审活动。

注 1：制定风险管理计划的指南见附录 F。

注 2：并非计划的所有部分都需要同时制定。可以随着时间的推移制定计划或计划的一部分。

注 3：风险的可接受性准则对于风险管理过程的最终有效性是至关重要的。对于每个风险管理计划，制造商宜选择适当的风险可接受性准则。

此外，选择可包括：

——在矩阵中（如图 D.4 和图 D.5）指出，哪一个伤害概率和伤害严重度的组合是可接受的或不可接受的；

——进一步细分矩阵（例如细分为可忽略的风险、可接受的已最小化的风险），并要求在确定风险可接受之前，先将其降低至合理可行的最低水平（参见 D.8）。

不论是何种选择，宜按照制造商确定风险可接受性准则的方针来决定，并基于适用的国家或地区法规以及相关的标准，而且要考虑可用信息，例如普遍接受的最新技术水平和已知的利益相关方的关注点（见 3.2）。建立此项准则的指南参见 D.4。

如果在医疗器械的生命周期内计划有所改变，应将更改记录保持在风险管理文档中。

用查看风险管理文档的方法检查符合性。

风险管理计划宜通过下列各项具体阐述软件是医疗器械的一部分：

——医疗器械的描述，包括医疗器械的何种功能将由软件实现；

——软件将依据 YY/T 0664—2008 开发的声明；

——引用软件开发方面所特有的软件风险管理(见注)；

——软件导致或软件控制的风险的可接受准则(如果与医疗器械其他组件的可接受准则不同时)。

注：引用软件开发计划也许是表明包含软件开发方面特有的软件风险管理的最简单方式。参见 3.4.2 和 3.4.3，其讨论了风险管理计划和软件开发计划的关系及根据 YY/T 0664—2008 标准的软件开发计划中与风险相关的特定主题。

软件导致或软件控制的风险的可接受准则可能与其他组件的风险可接受准则不同，一个原因是其伤害的概率不能估计。在这种情况下，风险可接受准则宜基于伤害的严重度。(关于软件导致的伤害的概率讨论见 4.4.3)。如果能断定危险(源)的实际后果很小，风险可以判定为可接受，不必要有风险控制措施。然而，对于重大危险(源)，即能造成严重度高的伤害的危险(源)，任何暴露水平都不能被认为其相应的风险低至可接受。在这种情况下，需要实施风险控制措施。

当概率无法估计时，剩余风险的可接受准则宜考虑已实施的风险控制措施和那些为降低伤害发生概率的风险控制措施的有效性。风险控制措施宜是所有合理可行措施的组合，满足适用标准和法规，并符合最新技术水平(见 YY/T 0316—2016 附录 D.4)。

当策划有关生产和生产后信息的收集和评审活动时，宜考虑以下对软件的特定要求：

——如果使用未知来源软件(SOUP)，宜计划积极监控和评价关于 SOUP 实际性能的可公开获得的异常清单及信息。如可能，在获得 SOUP 的同时，宜与 SOUP 提供者达成协议以支持以上活动。如果医疗器械的使用者可(有意地或无意地)自己修改(例如 SOUP 补丁或升级包)医疗器械的 SOUP，那么宜特别关注并监控市场上提供的新版本 SOUP(关于 SOUP 和生产后监视见第 9 章)；

——制造商宜使投诉者有可能识别和报告软件的版本。

3.4.2 风险管理计划与软件开发计划的关系

YY/T 0316—2016 对风险管理计划的要求和 YY/T 0664—2008 对软件开发计划的要求不宜被认为是要求具有特定标题的特定文档。计划的内容可以包含在任何文档中以适合制造商的质量管理体系，只要：

——计划的多个文档的结合宜以可证实的方式满足两个标准的要求；

——所有的计划宜彼此一致；

——所有的计划宜能及时提供使用；

——所有计划宜保持更新以反映变化的环境。

3.4.3 软件开发计划(根据 YY/T 0664—2008)的风险相关主题

软件开发计划宜确保软件开发过程、标准、方法和与软件开发相关的工具(依据 YY/T 0664—2008 的第 5 章在软件开发计划中描述)都是有效的风险控制措施(关于过程作为风险控制措施的讨论见 6.2.2.6)。这可以通过其他组织、供方和组织内的其他项目提供证据来实现。否则，就在本项目内策划并验证其有效性。

当建立医疗器械风险管理过程时，宜考虑软件风险管理的特有方面，如安全编码标准、验证方法(如形式化证明、同行评审、走查、仿真等)，以及语法和逻辑检查器的使用。如果考虑将这些要求作为风险控制措施，宜对其进行验证(参见表 B.2 中验证风险控制措施的示例)。

适当时，宜在计划、程序或培训中对医疗器械开发每个阶段的软件风险管理活动进行说明。

3.5 风险管理文档

YY/T 0316—2016 原文

3.5 风险管理文档

对所考虑的特定医疗器械，制造商应建立和保持风险管理文档。除本标准其他条的要求外，风险管理文档应提供对于每项已识别危险(源)的下列各项的可追溯性：

——风险分析；

——风险评价；

——风险控制措施的实施和验证；

——任何一个或多个剩余风险的可接受性评定。

注 1：构成风险管理文档的记录和其他文件，可以作为要求(例如制造商质量管理体系要求)的其他文件和文档的一部分。风险管理文档不需要包括所有的记录和其他文件。然而，至少宜包括所有要求文件的引用或提示。制造商宜能够及时地搜集到在风险管理文档中引用的资料。

注 2：风险管理文档可以使用任何形式或类型的媒介。

软件过程宜建立体系以使得以下的可追溯性是可能的，即从软件相关的危险(源)和软件风险控制措施以及追踪其实施，直至与安全有关的软件需求相对应，以及软件项满足那些需求的可追溯性。

上述所有活动宜验证其可追溯性(见 YY/T 0664—2008，7.3.3)。

因为软件在开发过程中可能频繁变更，并且发布不同版本，与软件相关的那部分风险管理文档也可能变更和有多个版本。

表 1 列出除 YY/T 0316—2016 的要求外，宜包括在风险管理文档中的 YY/T 0664—2008 关于文件的要求。

表 1 除 YY/T 0316—2016 的要求外，宜包括在风险管理文档中的文件的要求

YY/T 0664—2008 的条	风险管理文档文件
4.3c)	赋予每个软件系统的软件安全级别
4.3f)	对于不执行安全相关功能的软件系统中的软件项采用较低软件安全级别(相比软件系统)的理由的说明
7.1.4	软件项促成危险情况的可能原因
7.1.5	可能导致 YY/T 0664—2008 的 7.1.2 中所识别的危险情况的事件序列
7.2.1	对于软件项促成危险情况的每个潜在原因，规定风险控制措施
7.3.2	如果风险控制措施作为软件项实施，制造商宜评价该风险控制措施，以识别可能导致危险情况的任何新的事件序列，并形成文档
9.5	制造商宜保持问题报告及其解决情况(包括对其验证)的记录。适当时更新风险管理文档

4 风险分析

4.1 风险分析过程

YY/T 0316—2016 原文

4.1 风险分析过程

应按 4.2～4.4 中的描述针对特定的医疗器械进行风险分析。风险分析活动计划的实施和风险分析的结果应记录在风险管理文档中。

注 1：如果有类似医疗器械的风险分析或者其他相关信息可获得时，则该分析或信息可以用作新分析的起始点。这种相关程度取决于医疗器械之间的差别，以及这些差别是否会造成新的危险(源)，或者造成输出、特性、性能或结果的重大差异。对于已有分析的利用程度，也基于变化部分对危险情况形成的影响的系统性评价。

注 2：在附录 G 中描述了若干风险分析技术。

注 3：在附录 H 中给出了体外诊断医疗器械风险分析技术的附加指南。

注 4：在附录I中给出了毒理学危险(源)风险分析技术的附加指南。

除了 4.2～4.4 中要求的记录以外，风险分析实施和结果的文件还应至少包括：

a) 描述和识别所分析的医疗器械；

b) 识别完成风险分析的一个或多个人员和组织；

c) 风险分析的范围和日期。

注 5：风险分析的范围可以非常宽泛(如对于新医疗器械的开发，制造商知之甚少或没有经验)，或将风险分析范围进行限制(如分析更改对现有器械的影响，有关该器械的更多信息已经存在于制造商的文档中。)

用查看风险管理文档的方法检查符合性。

如 YY/T 0316—2016 所述，风险分析包含三个明确的活动：

——预期用途的识别；

——已知或可预见危险(源)(及其原因)的识别；

——估计每个危险(源)和危险情况的风险。

必须认识到风险分析是整个软件开发过程的不可或缺的一部分——不是一个或两个独立的事件，因为危险(源)和失效模式信息是在软件开发生命周期过程中产生的，需要在设计的每个阶段考虑风险分析以保证其有效。

因为很难对可能促成危险情况的软件异常的概率进行估计，软件方面的风险分析关注对软件潜在功能度和可能导致危险情况的异常的识别，而不是估计概率(关于估计概率的更多详细描述见 4.4.3)。

软件因素所能导致的最坏情况的伤害的严重度是一个确定软件开发过程的严格程度的基本输入(见 YY/T 0664—2008，4.3)。4.2，4.3，4.4 提供的信息旨在帮助识别有效的风险管理过程中软件的特定要求。另外，软件方面的风险分析宜可在结果文档中识别，且宜包括用于硬件失效的软件风险控制措施和导致危险(源)的软件原因，及其相关的风险控制措施。

4.2 医疗器械预期用途和与安全有关特征的识别

4.2.1 总则

YY/T 0316—2016 原文

4.2 医疗器械预期用途和与安全有关特征的识别

对所考虑特定的医疗器械，制造商应将预期用途以及合理可预见的误用形成文件。制造商应识别可能影响医疗器械安全的定性和定量特征并形成文件，适当时，规定界限。该文件应保存在风险管理文档中。

注 1：在文中，误用意指医疗器械的不正确或不适当的使用。

注 2：附录 C 包括了那些与用途有关的问题，可以用作识别影响安全的医疗器械特征的有用指南。

用查看风险管理文档的方法检查符合性。

每个医疗器械有其预期用途，宜考虑到对这些用途存在有意或无意误用的可能性。这不只是由软件引起的，但软件的使用可导致误用风险的增加，因为：

——医疗器械的运行状况更加复杂，因此掌握或理解更加困难；

——使用者可能对软件过分依赖，而不理解其局限性；

——医疗器械是可配置的，而使用者可能没有注意到当前配置；

——医疗器械可能需要和其他医疗器械或非医疗器械进行通信，而医疗器械制造商无法在细节上预见这种通信。

系统需求的提出者和软件工程师有一个共同的责任——将包括软件在内的系统的预期用途以及与安全和安全使用相关的所有系统和软件需求记录在风险管理文档中。软件工程师具体负责识别在系统水平上过于细微而不明显的预期用途要求。

4.2.2 用户界面

软件使设计更灵活的用户界面成为可能，这可能影响用户的行为，导致新的合理可预见的误用形式。通常的误用是源于对过于复杂的用户界面的误解，或为避免错误和不安全状态而过分依赖软件。要预见这些误用并且改变设计以尽可能的避免这些误用是很重要的。

这包括实施多语言的标记，特别是当这种标记是风险控制措施时。以下几点宜特别注意：

a) 不同语言需要的存储空间大小不同；

b) 不同字符集的使用；

c) 使用字符代替符号；

d) 使用不同的单位可能需要对数据结果再进行换算；

e) 日期格式和数字标点；

f) 不同语言和/或字符集的不同布局要求；

g) 确认的支持。

可用性过程见 IEC 62366[5]作为对 ISO 14971 的补充。

4.2.3 医疗器械的相互连接

医疗器械软件的使用使医疗器械和非医疗器械之间在一定范围内的相互连接和相互通信成为可能。这些连接和通信很可能引起由医疗器械和连接器械组成的系统产生新的应用(和误用)。虽然容易预见到可能发生这种新的应用和误用，但如果这些连接和通信不受限制，医疗器械制造商就不易识别所有这种应

用和误用。

因此，制造商对医疗器械的通信接口规定有限的预期用途很重要，并且在设计接口时尽可能限制连接和通信的做法是安全的。

例如，软件利用医疗器械内置接口检查治疗数据的一致性和合理性，这种检查基于对用户、患者以及数据产生前后关系的一致性。如果数据是由外部产生的，通过网络连接输入到医疗器械中，则可能无法进行同样的检查。在这种情况下，制造商要考虑如何把这种软件检查作为一种网络应用，使之对网络用户可用，和/或将数据输入限制在可信任的数据源，并且为在临床环境中负责网络连接的人员编写综合性的手册。

IEC 80001-1[6]包含了临床环境下医疗器械与 IT 网络的集成，其具体描述了制造商和将医疗器械接入 IT 网络的人员的责任。

4.3 危险(源)识别

YY/T 0316—2016 原文

4.3 危险(源)的识别

制造商应编写在正常和故障两种条件下，与医疗器械有关的已知和可预见的危险(源)文件。该文件应保存在风险管理文档中。

注：可能的危险(源)的示例在附录 E.2 和 H.2.4 中列出，可用作制造商启动危险(源)识别的指南。

用查看风险管理文档的方法检查符合性。

危险(源)识别的目的是分析所有可预见的危险(源)，并设计和实施有效的风险控制措施。

与热能、电能或者悬挂物不同，软件本身不是危险(源)(一种潜在的伤害源)；与软件接触不会造成伤害。然而，软件可能导致人体暴露在危险(源)下，换句话说，它可能促成危险情况。软件失效(任何形式)常常促进危险(源)转化为危险情况。

因而虽然软件很少引入新的危险(源)，但软件经常改变危险情况。对制造商而言，更重要的是避免危险情况的责任可由用户转移到制造商。

例如，解剖刀会产生显见的切伤危险(源)。然而，对这种危险(源)制造商按惯例不承担人体工学设计以外的责任，因为这种危险(源)被假定完全在外科医生的控制范围内。如果解剖刀是远程外科手术系统的一部分，同样的危险(源)仍然存在，但现在提供解剖刀控制软件的制造商要分担避免切伤危险(源)的责任。

这意味着一些在没有软件时仅依靠器械的专业使用来进行风险控制的危险(源)，现在转为依靠制造商的软件风险管理来控制。

一个重要的案例是由数据误处理带来的误诊的危险(源)。这的确是一种危险(源)，但当这些数据由临床医生处理时，制造商就没有责任。现在许多医疗器械利用软件来生成、存储、处理或使用数据，这使得危险(源)部分地成为制造商的责任。

软件促成危险情况有几种方式，包括以下几种(参见附录 B)：

——软件会如实的执行不安全的系统需求，导致具有危险性质的行为直到实际伤害发生才被察觉；

——软件规范可能错误地执行系统需求，导致非预期的行为，虽然这种行为符合软件规范；

——软件的设计和执行可能出错，导致与软件规范相反的行为。对软件规范的误解和将规范转换为代码时出现的差错会产生明显的缺陷。而软件项间和软件与底层结构间的非预期交互，可能导致较不明显的缺陷，这些底层结构包括硬件和操作系统。

对包含软件的医疗器械，进行仔细和全面的危险(源)识别可带来(在风险管理过程的后期)以下重要的结果：

——防止软件导致伤害的硬件风险控制措施；

——将潜在有害的软件功能从设计规范中排除；

——使用软件来防止伤害的风险控制措施(见 YY/T 0664—2008,5.2.3)；

——识别软件中必须以低错误率执行的部分和软件规范中必须进行特殊测试的部分(YY/T 0664—2008,4.3)；

——识别较高的安全级别的软件项,其应与其他软件项(较低的软件安全级别)隔离以防止非预期负面影响产生的伤害(见 YY/T 0664—2008,4.3 和 5.3.5)。对此的进一步讨论见 6.2.2.2.4。

为充分识别危险(源),应对医疗器械的临床使用进行很好的理解。另外,软件的复杂性带来了特别的挑战,包括了可能的复杂用户界面。因此,软件危险(源)识别不能孤立地进行,其宜由多学科的团队从系统层面去开展,团队包括临床专家(如临床支持和技术服务专家),软件工程师,系统设计师,可用性/人因工程专家(见 3.3)。

危险(源)识别宜考虑医疗器械的性质可能导致的伤害(如切伤、辐射或者电死患者)及与软件使用有关的额外的危险(源)。后者可能包括：

——向临床医生或患者提供错误信息；

——对患者的错误识别(在医疗器械存储患者详情和处方的情况下)；

——由软件异常引起的治疗延误或治疗无法进行。

注：对许多医疗器械来说,治疗延误或治疗无法进行不被认为是对患者的伤害。

识别的危险(源)宜包括按照其规范操作的与软件有关的危险(源)及与软件异常有关的危险(源)(见 6.1)。

通常软件会使医疗器械的用户界面更复杂。特别是包括软件的医疗器械需要经常处理信息,可以根据对患者是否有益来判别这种复杂是否合适,宜考虑与错误信息及错误使用信息相关的额外危险(源),例如：

——错误的数据输入；

——引起用户误读的显示；

——用户对报警的误解和忽视；

——由于数据或报警过多而导致的用户超负荷(见 IEC 62366[5])。

4.4 估计每个危险情况的风险

4.4.1 总则

YY/T 0316—2016 原文

4.4 估计每个危险情况的风险

应考虑可能造成危险情况的合理可预见的事件序列或组合,造成的一个或多个危险情况应予记录。

注 1：对事先不能识别的危险情况,可以使用覆盖特定情况的系统性方法(见附录 G)。

注 2：在 H 2.4.5 和 E.4 中给出了危险情况的示例。

注 3：危险情况可能由疏忽、失误和差错造成。

对每一个已识别的危险情况,都应利用可获得的资料或数据估计与其相关的一个或多个风险。对于伤害发生概率不能加以估计的危险情况,应编写一个可能后果的清单,以用于风险评价和风险控制。这些活动的结果应记录在风险管理文档中。

任何用于对伤害的发生概率和伤害的严重度进行定性或定量分类的体系,都应记录在风险管理文档中。

注 4：风险估计包括发生概率和后果的分析。按照应用情况，只有风险估计过程的某些要素可能需要考虑。例如在有些情况下，不需要超出初始危险(源)和后果分析的范围。见 D.3。

注 5：风险估计可以是定量的或定性的。附录 D 中给出了风险估计的方法(包括那些由系统性故障产生的风险)。附录 H 给出了体外诊断医疗器械风险估计的有用资料。

注 6：用于风险估计的资料或数据，可由如下方面获得：

a) 已发布的标准；

b) 科学技术资料；

c) 已在使用中的类似医疗器械的现场资料(包括已公布的事故报告)；

d) 由典型使用者进行的可用性实验；

e) 临床证据；

f) 适当的调研结果；

g) 专家意见；

h) 外部质量评定情况。

用查看风险管理文档的方法检查符合性。

要估计与软件有关的风险，首先应对包括软件在内的危险情况进行识别。软件既可能是导致危险情况的事件序列的初始原因，也可能是该事件序列中的其他原因，比如对于预期检出硬件故障的软件就是这种情况。软件可能包含未知来源软件(SOUP)组件或重新使用以前开发过的组件。

风险估计是基于伤害的概率和每一个已识别的危险情况所导致的伤害的严重度。因为很难估计软件异常引发伤害的概率(见 4.4.3)，所以在估计危险情况(导致伤害的事件序列中包含软件异常)的风险时，须谨慎使用软件异常发生的概率。

4.4.2 识别的方法

识别软件在危险情况中的潜在作用有多种方法。这些技术方法采用不同的手段，在软件开发的不同阶段发挥作用。其中的任何一种都不是唯一恰当的方法。关于风险分析的一些可利用的技术信息见 YY/T 0316—2016 附录 G。

故障树分析(FTA)是一种传统的自上而下的方法(参见 IEC 61025[3])，其通常从医疗器械整体开始分析。FTA 主要用于分析伤害的起因。其假定伤害发生，利用布尔逻辑来识别伤害发生必须的事件或条件，以逐步细化的方式分析事件或条件，直到识别一个或多个能防止伤害的风险控制措施。FTA 可以用于识别导致危险情况的事件序列中涉及的软件项。

失效模式和效应分析(FMEA)是一种自下至上的方法(见 IEC 60812[2])，其从组件或子系统(对软件来说就是 YY/T 0664 中的软件项)开始，并提出问题：如果该要素失效，那后果是什么？

考虑到预见每个软件项中有哪些软件缺陷的难度，FMEA 的起始点是列出每个软件项的安全相关需求，并考虑这个问题：如果需求无法满足，后果是什么？

这导致识别哪些软件项的失效会引起伤害，以及识别需要防止哪些类型的失效。

当识别能导致危险情况的事件序列或事件组合时，关注直接与医疗器械基本性能相关的软件(例如计算血液葡萄糖水平的算法)和相关危险(源)的具体原因是最简单的。考虑软件原因可能导致不明显的失效模式并因此导致一个或多个医疗器械的危险(源)也很重要。软件原因的示例参见附录 B。

注：具体原因是指软件缺陷，其功能与器械临床功能明显相关，并是导致器械危险(源)的原因之一。例如，计算试验结果的算法缺陷。

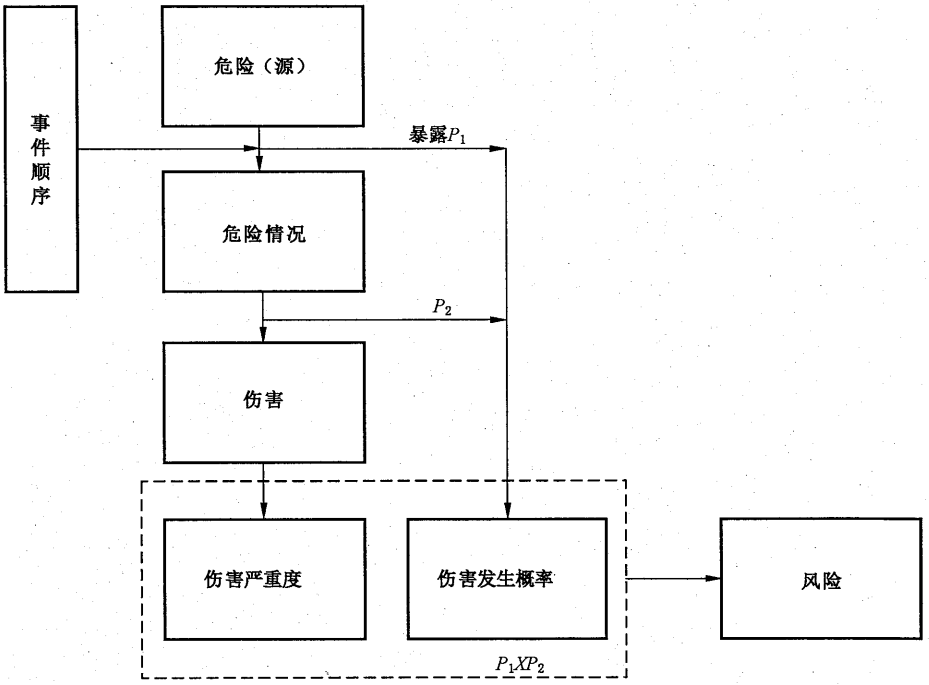
精确的预见在软件项中发生什么失效是困难的，但识别缺陷的类别是可能的，每一类都有众所周知的风险控制措施。例如，数据破坏类故障可以通过求和校验程序来检出和防止。软件原因示例及其建议处理方式参见附录 B。制造商宜保持与其产品相关的软件缺陷类别清单。

4.4.3 概率

特定版本软件的异常在该软件的所有拷贝中都会出现。然而因为软件的每个单独拷贝输入的随机性，软件异常导致软件失效的概率很难估计。

没有一致认可的软件失效发生概率的估计方法。当软件出现在导致危险情况的事件序列中时，估计危险情况的风险时无法考虑软件失效发生的概率。在这样的情况下，考虑一个最不利的可能性是适当的，即软件失效发生的概率宜设成 1。当可以做到估计序列中其余事件(如果其不是软件)的概率时，该概率可用作危险情况的发生概率(图 1 中的 P_1)。如果无法做到，危险情况的发生概率宜设为 1。

估计导致伤害的危险情况的概率(图 1 中的 P_2)通常需要临床知识，以区分临床实践很有可能防止伤害发生的危险情况和更有可能引发伤害的危险情况。



注 1：P_1——危险情况发生概率。

注 2：P_2——危险情况导致伤害的概率。

图 1 危险(源)、事件序列、危险情况和伤害之间的关系示意图—取自 YY/T 0316—2016 附录 E

在许多情况下，无法估计伤害发生概率，只能依据伤害的严重度估计风险。在这些情况下的风险估计宜关注危险情况引发的伤害的严重度。

虽然可能无法估计软件失效发生的概率，但很明显许多风险控制措施降低了这种失效导致危险情况的概率。例如，软件异常导致的内存崩溃。存储的求和校验会检出失效并降低危险情况的概率。该求和校验不能保证所有可能的崩溃都被检出，然而，这种手段将检出绝大部分的崩溃并因此把风险降低到可接受水平。虽然在实施求和校验之前或之后都无法估计危险情况发生的可能性，但可以断言，求和校验执行后，危险情况发生的可能性比实施求和校验之前降低了。制造商有责任证实风险控制措施对于已识别的剩余风险满足在风险管理计划中的可接受性准则是有效的。

总之，软件风险估计宜主要关注严重度和如果失效发生引发伤害的相对概率，而不是试图估计每个可

能的软件失效的概率。

注：这有助于区分相同严重度的危险(源)，以便更加关注那些更有可能引发实际伤害的危险(源)。

4.4.4 严重度

对软件导致的风险的严重度估计，影响所采用的软件开发过程。根据 YY/T 0664—2008，过程的严格程度取决于软件可能引发的伤害的严重度。

为了按照 YY/T 0316—2016 进行风险评价，制造商更愿意接受如何定义严重度等级的方法，与 YY/T 0664—2008 中软件安全分类相联系并有助于定义严重度等级。否则可能需要两次定义严重度，一次为了整个风险管理过程所需的风险评价，另一次为依据 YY/T 0664—2008 确定软件安全级别。

5 风险评价

YY/T 0316—2016 原文

5 风险评价

对每个已识别的危险情况，制造商应使用风险管理计划中规定的准则，决定是否需要降低风险。如果不需要降低风险，则 6.2～6.6 给出的要求不再适用于此危险情况(即前进到 6.7)。风险评价的结果应记录在风险管理文档中。

注 1：附录 D.4 中给出了风险可接受性的决策指南。

注 2：应用相关标准作为医疗器械设计准则的一部分，可以构成风险控制活动，以满足 6.3～6.6 中给出的要求。

用查看风险管理文档的方法检查符合性。

如 4.4.3 中所述，很难估计软件失效的概率。当不能估计伤害的概率时，那么风险估计只基于伤害的严重度。

6 风险控制

6.1 降低风险

YY/T 0316—2016 原文

6 风险控制

6.1 降低风险

当需要降低风险时，应按照 6.2～6.7 的描述执行风险控制。

软件方面的风险降低，在 6.2～6.7 中讨论。

6.2 风险控制方案分析

YY/T 0316—2016 原文

6.2 风险控制方案分析

制造商应识别适于将风险降低至可接受水平的一个或多个风险控制措施。

制造商应按下列顺序，依次使用一种或多种方法：

a) 用设计方法取得固有安全；

b) 在医疗器械本身或在制造过程中的防护措施；

c) 安全信息。

注 1：如果实施方案 b)或 c)，在决定风险是否可接受之前，制造商可遵循一个过程：考虑合理可行的风险控制措施，并选择适当的风险降低方案。

注 2：风险控制措施可以降低伤害的严重度，或减少伤害的发生概率，或两者都减少。

注 3：许多标准为医疗器械阐述了固有安全、防护措施和安全信息。此外，许多其他医疗器械标准融入了风险管理过程的要素(例如电磁兼容性、可用性、生物相容性)。相关标准宜用作风险控制方案分析的一部分。

注 4：对于不能估计其伤害发生概率的风险，见 D.3.2.3。

注 5：附录 J 中提供了安全信息指南。

所选择的风险控制措施应记录在风险管理文档中。

如果在方案分析中，制造商确定所需的风险降低是不可行的，制造商应进行剩余风险的风险/受益分析(进入 6.5)。

用查看风险管理文档的方法检查符合性。

6.2.1 复杂系统风险控制方案的选择

6.2.1.1 总则

在复杂系统中，可能有许多能导致危险情况的事件序列。不可能或没必要对这些序列中的每个事件都应用风险控制措施。有选择地对一些事件应用风险控制措施，并且把总的伤害概率降至可接受水平就足够了。

在以下三条中，给出如何在软件中应用三种风险控制措施的概述。另外还讨论了哪些事件需要风险控制措施(见 6.2.1.5)。

6.2.1.2 用设计方法取得固有安全

通过设计的方法取得固有安全常常通过去除医疗器械的不安全特性来实现，或者通过更改设计用更安全的方法(即避免或最小化危险情况的方法)来实现特性。这通常有简化设计的效果，使设计实现和用户操作更加容易。

这对于通过软件实现的特性更是如此。在开发软件系统时，不加辨别地包含所有可能的客户愿望是一种诱惑。这可能导致软件组件之间交互方式的大量增加，带来不期望的危险情况。通过在医疗器械及其软件的早期开发过程中实施风险管理，可以避免这些问题并且使大部分客户满意。

在大多数情况下，用设计方法取得固有安全(对于软件)包括：

——除去不必要的特性；

——更改软件体系结构以避免导致危险情况的事件序列；

——简化用户界面以降低使用中人为错误的可能性；

——规定软件设计规则以避免软件异常。

后者包括：

——仅使用静态内存分配以避免动态内存分配带来的软件异常；

——规定使用编程语言的版本以避免很可能导致的编程错误。

6.2.1.3 防护措施

使用软件的医疗器械的防护措施可通过硬件或软件实现。防护措施的设计宜证明防护措施独立于其所应用部分的功能。如果在硬件上应用软件防护措施，这是容易实现的，反之亦然。

选择以软件的方式实现防护措施并应用于软件时，避免一个原因导致多个失效的可能性是重要的。如果一个防护措施检出和/或防止了一个危险情况，制造商宜证明该防护措施和提供基本性能的软件功能具有充分的隔离。

例如，为患者提供治疗的软件在一个处理器上运行，而实现软件防护措施的软件在另一个独立的处理器上运行。

6.2.1.4 安全信息

医疗器械软件的使用有可能导致如用户所看到的更复杂的行为。这有可能导致对安全信息更多的依赖，从简单的屏幕警告到复杂的用户手册和明确的培训课程。注重良好的用户界面设计可以降低这些书面材料的量和复杂程度(见 IEC 62366[5])。

6.2.1.5 哪些事件需要风险控制措施

许多事件序列都可能导致危险情况。不可能或没必要对这些序列中的每个事件应用风险控制措施。有选择的对一些事件应用风险控制措施，并且把总的伤害概率降至可接受水平就足够了。

在决定哪些事件宜被检出、防止或降低发生的可能性时，绘制出能导致危险情况的事件序列显然是有帮助的。虽然故障树分析(见 4.4.2)不显示事件序列，但可以通过它来识别这些序列。风险控制措施的正确操作就像是“与门”的一个“非”输入，不管“与门”其他的输入是什么，伤害都会被阻止。图 2 表示 FTA 图的一部分，图中一个不正确的软件输出(其本身是一个事件序列的输出)被风险控制措施检出不安全情况并且防止输出有害影响(例如，通过停止一个动作)，进而防止对患者的伤害。只有当风险控制措施失效时，不正确的软件输出才能伤害患者。需要注意的是，导致不正确软件输出的事件序列并不需要探究细节以确保其不能导致患者伤害。

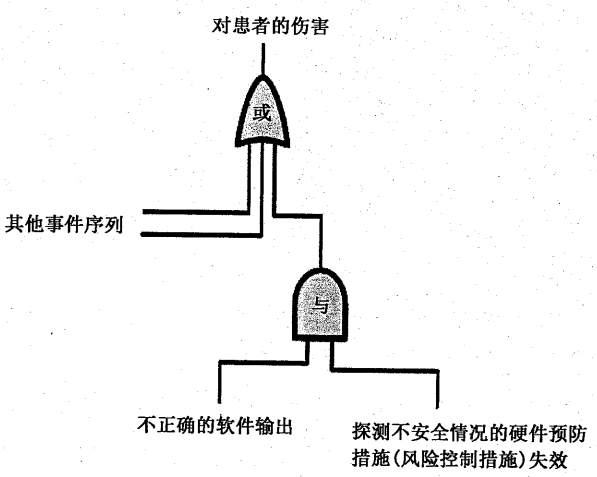


图 2 风险控制措施阻止不正确软件输出引发伤害的 FTA 图示

明显可以应用风险控制措施的点包括：

——软件系统的输入作为一个整体；

——软件系统的输出作为一个整体；

——软件模块之间的内部接口。

限制软件输入范围的风险控制措施能防止不安全的输出。它也可以降低输入导致伤害(由于软件异常)的可能性(见4.4.3)，因为其降低了软件以可能未经测试的非预期方式运行的可能性，但这种降低不是太明显。

限制软件输入范围可以通过软件或硬件风险控制措施来实现，例如：

——能检查输入并且拒绝不安全或不一致值的软件风险控制措施；

——硬件风险控制措施可能包括一个上锁的房间以防止未经授权的人输入数据。

设置于医疗器械或其软件的风险控制措施输出可以检查软件输出值是否在安全范围内和是否具有内部一致性以防止伤害发生，这可以通过以下方法实现，例如：

——能检查输出值并防止其超出安全范围的软件风险控制措施；

——限制应用于患者的能量的硬件风险控制措施；

——由警告标签和电路连接的“停止”开关组成的风险控制措施。这种风险控制措施假定有能力的操作者能检出危险情况。

除了可以对医疗器械或其软件的输入和输出实施风险控制措施外，也可以对软件组件的输入和输出实施风险控制措施。这样可以检查软件更小组件的输入和输出并且防止伤害。

可能无法规定医疗器械安全运行的单一参数范围，然而可以规定一个“安全操作范围”，换句话说一个组成医疗器械安全运行边界的参数组合。软件可以用来评定医疗器械的操作是否在安全操作范围内。例如，软件可以通过对应用部分测得的温度和暴露时间的组合来检出灼伤患者的可能性。

在有些案例中，临床医生知道软件输入和输出的安全范围，但在医疗器械的设计中却无法预期。在这些案例中，可以通过风险控制措施确保医疗器械严格按照临床医生规定来运行。可以用硬件或软件的风险控制措施来检出软件输出和输入的不一致。

例如，临床医生在使用医疗器械时，可对不同的患者开出不同的治疗处方。只通过分析输入值和输出值无法检出危险情况。尽管如此，应用软件风险控制措施可确保医疗器械的输出与输入(预期处方)的精确匹配。

6.2.2 风险控制方法

6.2.2.1 概述

为了对软件有效地实施适当的风险控制措施，宜深入理解产品开发和软件生命周期。一些类型的风险控制措施在设计早期很容易实施，但在开发过程晚期不可能实现或成本太高。如果在产品开发过程的早期没有从风险管理角度仔细考虑软件，可能要采取硬件方面的措施，解决过分依赖软件正常运行来确保医疗器械的安全。

把软件项分隔开很有用，为不同的软件项分配不同的软件安全级别，这样可以区分关键软件项(例如，缺陷可以导致死亡的)和不会影响安全的软件项(软件的安全级别见YY/T 0664—2008，4.3)。

确定软件安全级别，可以作为对多数关键软件项采取更严格控制、更重点验证和配置管理活动的基础。如果这样做，宜仔细考虑其副作用，对较低关键的软件项宜与受其影响的任何多数关键软件项同样评级。需要注意YY/T 0664—2008允许在同一个活动或任务中使用不同的方法(参见YY/T 0664—2008，5.1.4)。制造商可以制定方案来区分软件安全分类为C级的软件项。例如，制造商可对复杂程度高的软件项使用更正式的验证方法(例如，相对于代码评审而采用代码检查)。

需要注意的是，软件项可能在初期被定为安全相关，然后通过选择某些风险控制措施或设计，软件项

可作为较低关键项对待。合理地执行风险管理可以通过隔离和固有安全设计方法，使与安全相关的软件子集尽可能减少到最小。

确保软件安全需要贯穿产品开发生命周期的各种活动。可靠性技术(例如正式的故障分析方法)并不能构成完整的风险管理方法。还有一点认识也非常重要，可靠性和安全，尽管非常相关，但是它们不是同一个属性。关注软件生命周期过程的可靠性并不一定能达到足够的安全。

6.2.2.2～6.2.2.6 更详细描述了一些具体的风险控制措施，并对风险的具体原因如何处置给出了指南。

6.2.2.2 风险控制措施和软件结构性设计

6.2.2.2.1 概述

软件体系结构宜描述通过固有安全设计方法来控制风险的软件特性，以及通过防护措施来降低风险的软件机制。

6.2.2.2.2 通过体系结构特性提供的固有安全设计

与软件控制功能相关的危险(源)可以避免，例如通过硬件实现功能。类似地，与硬件功能相关的危险(源)(磨损，疲劳)可以通过使用软件来避免。

有时某个危险(源)可以通过高层级的设计决策完全避免。例如，从硬件的方面看，使用电池代替交流电源可以完全避免触电死亡的风险。同样的，导致危险(源)的整个编程错误可以通过高层级的设计决策来消除。例如，内存溢出可以通过仅使用静态数据结构来避免。

在应用软件的系统中有个特殊的问题，认为软件可以不受限制地分享物理结构。这种理解是错误的。

系统设计中有一个通用规则，系统宜包括足够的资源，以便需要时能执行所有的任务。这个规则对软件和硬件都适用。如果一个软件项和安全相关，风险评定宜关注以下问题：

——这个安全相关软件项需要时能否访问它的处理器；

——在不安全状况发展成事故前，安全相关软件项能否确保足够的处理器占用时间以完成其任务；

——能否证明其他软件项不会破坏或干扰安全相关软件项。

如果安全相关软件需要和非安全相关软件共享处理器，那么以上的问题尤为重要，因为安全功能会和非安全功能竞争资源(关于隔离参见 6.2.2.2.4)。

选择开发方法时宜确保设计者可得到以上问题。例如，安全相关软件项设计成当操作系统空闲才能运行是不够的。开发方法宜当支持计划安排、优先级和时序的审慎设计。

6.2.2.2.3 容错体系结构

为了确保患者或用户的安全，需要医疗器械有许多功能。这些功能包括不能中断或延迟的临床功能和实现防护性风险控制措施的功能。

容错设计是一种非常通用的方法，用以提高医疗器械可信性(参考软件工程实践者，包括 Pullum[7]和 Banatre[8])。容错设计的目的是确保安全相关的功能在组件故障(包括软件异常)发生时仍然继续运行。

容错设计常常利用冗余。冗余可能利用至关重要的组件的一个简单重复，以便在一个组件失效时系统能继续运行，或者由附加组件组成，以便检出失效并切换到替代的运行模式中，这也许限制了部分功能。

容错设计能在软件失效时使至关重要的功能继续运行。在这种情况下，使用同一软件多重拷贝的简单冗余就不充分了，因为每个软件拷贝中都存在同样的缺陷。

在这种情况下，需要使用多样性。例如，使用附加的软件检出软件错误和完成恢复程序。附加的软件宜避免与被监控的软件共享同样的特性，以消除一个软件缺陷导致两个软件都失效的可能性。

在更关键的情况下，两个或多个软件项完成同样的功能，但他们从一个通用的规范开始，独立设计和执行。这就是“多样性编程”。然而需要注意，不同开发工程师有犯同样错误的趋势，这会使多样性失效。

还要注意通用规范可能包含不正确的要求。最终，要使用比如表决的方法来确保故障软件不会产生影响。至少需要三个不同的软件项来进行表决。

当使用冗余来实现容错设计时，不管是否使用多样性，告知用户失效发生很重要。否则，具有容错设计的医疗器械可能看起来运行正常，但实际上是在降低安全运行。

6.2.2.2.4 通过隔离降低软件原因带来的风险

软件缺陷可能使同一硬件中运行的不相关软件发生错误。制造商宜选择隔离安全相关软件项和非安全相关软件项的方法，使非安全相关软件项不能干扰安全相关软件项的运行(参见 YY/T 0664—2008，5.3.5)，并且证明隔离是有效的。这包括证明软件项使用合适的资源(物理或时间)以避免软件项间非预期的争夺。

软件项间的有效隔离应处理以下几种软件项发生非预期交互的可能方式：

- 当软件争夺共享硬件(例如处理器、存储设备和其他输入/输出设备)的占有时间时，软件项可能以非预期的方式互相影响。这会阻止软件项在预期的时间运行。提供充分的硬件资源是一种结构特性(参见 6.2.2.2.2)，宜有足够的规范和策划来确保需要时有充分的时间来运行所有的软件项。
- 多个软件项可能共同存在于同一个内存中；这会导致一个软件项非预期的更改另一个软件项的数据。在极端情况下，一个软件项可能偶然地更改了另一个软件项的编码。许多处理器和操作系统提供硬件辅助方法来隔离内存使用。当有这些方法时，宜使用这些方法。即使其中一个软件项存在缺陷，多数这样的方法会防止非预期的相互影响。
- 当软件项共享变量时，也会带来非预期的相互影响，包括全局变量，环境变量和操作系统参数；这会导致当一个软件项存在缺陷时，软件项间产生非预期的交流。软件项间共享变量宜做到最小化，必要时，向所有工程师发布规则，以确保只有少数规定的软件项可以更改共享变量，而其他的软件项只读取共享变量，而不能更改他们。

最强的隔离方式是在不同的处理器上运行不宜相互影响的软件项。然而，上述推荐的谨慎的结构设计可以在单一处理器上提供适当的隔离。

在实验室环境中测试系统对于给定的测试用例，可能物理和时间资源是充分的，然而现场的应用负荷或执行环境(其他过程在同一地点运行)使软件失效在某种程度上导致伤害。

另一方面，如果实验室的测试用例显示性能不好，并且采用无效的措施草率的加速软件，这些措施可能破坏设计并通过不可预见的副作用增加其他风险。

有效的隔离宜证明在正常操作下：

a) 能防止数据流崩溃：非安全相关软件项不能修改安全相关数据；

b) 能防止控制流崩溃：

——安全相关功能总是在正确的时间执行，不被非安全相关软件项的行为影响；

——非安全相关软件项不能修改安全相关软件项；

c) 能防止执行环境崩溃：不会发生安全相关和非安全相关软件项共用的软件系统(例如处理器、寄存器、设备寄存器、内存访问特权)部分崩溃。

导致违背上述原则的事件(例如硬件失效)宜能被检出并且使系统采取必要的措施来确保持久的安全。

6.2.2.3 防护性措施细节

在多数情况下，通过固有安全设计方法或者在所有可能失效的情况下执行容错，来避免所有危险(源)是不切实际的。在这些情况下，防护措施是次优的管理潜在危险(源)的方法。这些措施通常通过检出潜在危险情况来运行，既可自动干预以减轻后果，又可产生报警使用户干预。

例如，X 射线治疗系统有一个使用软件逻辑或硬件的互锁系统，当通往该设备的任何门打开时，关闭

X射线发生器。互锁功能对治疗没有作用。其唯一目的就是减轻非预期的射线暴露的伤害。

在某些情况下[如医疗器械功能丧失不会产生危险(源)],安全可能以未完成任务为代价获得。例如,实验室血液分析仪未能提供结果,在某些情况下这不会是危险(源),但提供错误结果可能是危险(源)。在这个示例里,当保护性检验程序显示非预期的故障时关闭分析仪,而不是继续工作,以降低风险。在故障-安全(*fail-safe*)体系结构中,系统或组件故障,或其他危险(源)条件,可导致丧失功能,但在某种程度上可保护操作者和患者的安全。在故障-操作(*fail-operational*)系统中,系统可继续安全操作,但性能降低(例如,降低容量或减慢响应时间)。

6.2.2.4 即时防止和公布危险情况

某一类重要的风险控制措施是提高防止危险情况的可能性的措施。

除了防止伤害,也宜考虑向用户公布检出的情况。如果不这样,后续的风险控制措施失效可能使伤害发生。

宜考虑软件风险控制措施的运行频率。软件风险控制措施宜运行的足够频繁以便在发生伤害之前检出危险情况。

6.2.2.5 软件异常的风险控制措施

软件存在一个难点:事件序列中导致危险情况的事件是由未发现的软件异常导致的,很难预期哪里会发生这些异常或这些异常的后果是什么。

风险控制措施能降低软件异常引发伤害的可能性。不管先前事件的性质怎样,通常软件体系结构中总存在着一些位置,可以施加风险控制措施以降低伤害的概率。如果谨慎的做到这些,就没必要通过准确预测软件异常的性质来防止他们引发伤害。

在这种方法不可行的情况下,例如在软件中实现防护措施不可行时,则宜使用确保软件完整性的方法(参见6.2.2.6)。

6.2.2.6 过程作为风险控制措施

如果软件异常可能促成导致危险情况的事件序列,可能无法设计防止伤害发生的风险控制措施。这种情况下,最好的解决办法是用设计方法取得固有安全来确保软件异常不可能导致危险情况。

当固有安全设计方法不可行时,可以利用有效的软件开发过程来降低软件异常发生的概率。一致的共识是,当与其他类型的风险控制措施一同考虑,并且详细定义时,过程风险控制措施是有益的方法。

如果能对软件可靠地完成预期功能建立高度信心,并且对软件无故障的信心很高,则软件可能被视为高完善性组件。为了达到这种高度信心,制造商应证明软件开发过程能预期产生高度可靠、无故障的软件。应用这样的过程则可以声称降低了软件异常发生的概率。

一般认为,提高软件开发过程的严格程度可以降低软件异常的数量。需要注意的是虽然医疗器械软件测试可以降低软件异常的数量,但不能假定当软件通过所有计划的测试时,就没有软件异常了。这是因为在临床使用中,软件的输入包含测试计划外的序列。由于医疗器械软件太复杂以致于不能无遗漏的测试,所以严格的测试只能看作是降低危险情况可能性的一种方法。然而测试本身,不足以建立软件可被视为高完善性组件的信心。

包含YY/T 0664—2008中规定的软件开发过程的活动和任务,可以作为定义严格软件开发过程的起始点。开展严格软件开发过程需要考虑的其他项目可包括:

a) 人员能力——技能、资质、经验和培训(谁开发软件?);
b) 方法——规范、设计、编码和测试方法的适用性(开发过程是什么?);
c) 严格的、正式的及评审和检查的范围(进行了多少静态分析?);
d) 工具——工具(诸如编译器、要求的可追溯性和配置管理工具等)的质量(软件开发过程中使用什

么工具?)。

想要开发高完善性软件取决于具有一个始终遵循的可重复的过程。

通过实施严格的开发过程来降低软件异常导致危险情况的风险时,宜通过收集和分析软件异常导致失效的频率的数据来证明风险控制措施的有效性。要声称某个过程产生了高完善性软件,宜有证据来支持没有或很少有软件失效。

6.2.3 对未知来源软件(SOUP)的考虑

使用 SOUP 的决定通常是在系统设计阶段做出的。医疗器械潜在风险越高,就越要对 SOUP 的潜在故障模式进行分析并确定风险控制措施。通常不能通过更改 SOUP 添加监控或隔离 SOUP 所需的新风险控制措施,以防止 SOUP 失效时引发危险(源)或危险情况。也没有足够的内部设计信息可以用来识别所有由于 SOUP 引起的潜在危险(源)。所以系统和软件体系结构设计时宜提供必要的风险控制措施监控或隔离 SOUP,以便防止 SOUP 发生故障时产生危险(源)。

当医疗器械包含 SOUP 组件时,应注意该软件不能危及医疗器械安全。这种情况可能需要引入软件"封装"或中间件体系结构。中间件可以:

a) 阻止不希望使用的 SOUP 特性的使用;

b) 执行逻辑检查以确保正确的信息在 SOUP 和医疗器械软件之间传送;

c) 提供医疗器械需要的附加信息。

还有一个与 SOUP 相关的关键问题就是商用操作系统和通讯系统的使用。宜建立一种软件体系结构,在不影响安全的同时进行全面的风险评定的基础上,允许对软件平台做一些更改(例如稳定性、保密安全)。这种风险评定宜包括必要的更改频次的分析以确保完整的医疗器械安全,例如安装网络保密安全补丁。

6.3 风险控制措施的实施

YY/T 0316—2016 原文

6.3 风险控制措施的实施

制造商应实施在 6.2 中选择的一个或多个风险控制措施。

每一项风险控制措施的实施应予以验证,此项验证应记录在风险管理文档中。

风险控制措施的有效性应予以验证,且验证结果应记录在风险管理文档中。

注:有效性的验证可包括确认活动。

用查看风险管理文档的方法检查符合性。

一旦确定了风险控制措施,那么这些措施就需要得到执行,并验证其有效性。

验证风险控制措施已正确实施并且对风险控制是有效的,这对软件来说是非常基本的。分析和测试很可能都是必要的。要考虑的关键方面包括:

a) 在所有相关版本和变量(例如不同的平台、语言和医疗器械型号)中,确保所有安全相关软件项被识别出来,确保所有安全相关的功能被规定、执行和测试的可追溯性;

b) 测试风险控制措施时更严格和更大的覆盖范围的测试,包括在大范围的异常和压力条件下的测试;

c) 当发生改变时,即使这些改变不预期影响安全,也要关注测试风险控制措施和与安全相关的功能的回归测试。

即使所用的过程被认为严格到足以产生高完善性的软件,结果仍应被验证。

如果进行了根本原因分析以及跟踪和评价了与安全相关异常的关键性评级(参见 YY/T 0664—2008,9.1),在验证和确认活动中收集的异常信息常常是有用的。可评价有安全影响的异常以确定在风险评定中

是否被识别，以及已识别的风险控制措施一旦实施是否充分。软件异常的数据可用来证明软件开发过程的有效性，或用来识别软件开发过程的哪些方面需要改进以降低风险。

软件风险控制措施的充分性可能不如硬件风险控制措施的充分性那样明显。因此当处理软件时，宜考虑风险评价的文档除传统形式外是否需要其他的形式。一个有用的方式是编写“安全用例”(参见附录 E)。

6.4 剩余风险评价

YY/T 0316—2016 原文

6.4 剩余风险评价

在采取风险控制措施后，对于任何剩余风险，都应使用风险管理计划中规定的准则进行评价。此项评价结果应记录在风险管理文档中。

如果剩余风险使用这些准则判断为不可接受的，应采取进一步的风险控制措施(见 6.2)。

对于判断为可接受的剩余风险，制造商应决定哪些剩余风险应予以公示，并且需要将信息包括在随附文件中，以便公示那些剩余风险。

注：在附录 J 中提供了如何公示剩余风险的指南。

用查看风险管理文档和随附文件的方法检查符合性。

软件产生的剩余风险需要包括在医疗器械系统级的剩余风险中。由于很难评估软件异常的概率，剩余风险评价包括：确定是否所有能导致不可接受风险的事件序列都有风险控制措施以降低发生的可能性，或将风险管理计划中定义的伤害的严重度降至可接受水平(参见 3.4.1)。

任何被识别出来的尚未纠正的软件异常(在验证和确认活动中)都宜通过分析来确定其是否影响安全相关软件(参见 YY/T 0664—2008，5.8.3)。如果影响，这些异常引起的风险需要评价，并用于他们可影响的任何危险情况的剩余风险的评价。

6.5 风险/受益分析

YY/T 0316—2016 原文

6.5 风险/受益分析

如果使用风险管理计划中建立的准则，判断剩余风险是不可接受的，而进一步的风险控制又不可行，制造商可以收集和评审资料和文献，以便决定预期用途的医疗受益是否超过剩余风险。如果此项证据不支持医疗受益超过剩余风险的结论，则剩余的风险是不可接受的。如果医疗受益超过剩余风险，则进行 6.6。

对于经证实已被受益超过的风险，制造商应决定哪些安全信息对公示剩余风险是必要的。

评价结果应记录在风险管理文档中。

注：见 D.6。

用查看风险管理文档的方法检查符合性。

对软件没有补充指南。

6.6 由风险控制措施产生的风险

YY/T 0316—2016 原文

6.6 由风险控制措施产生的风险

应对风险控制措施的有关以下方面的影响进行评审：

a) 新的危险(源)或危险情况的引入；

b) 是否由于风险控制措施的引入，影响了对以前识别的危险情况所估计的风险。

对任何新的或增加的风险应按照 4.4～6.5 进行管理。

评审结果应记录在风险管理文档中。

用查看风险管理文档的方法检查符合性。

严格的软件配置管理过程，包括严格的更改控制(参见 YY/T 0664—2008，8.2)是必须的，以便所引入的软件风险控制措施对医疗器械其他部分的影响得以仔细的检查(参见 YY/T 0664—2008，7.4)。

YY/T 0316—2016 并未规定设计和开发过程。这样的一个结果就是：当风险控制措施已执行，YY/T 0316—2016 只是简单地要求对这个措施进行评审以确保不产生进一步的危险(源)。这不宜理解为仅在实施完风险控制措施后才要求检查问题。

对风险控制措施，特别重要的是不能直到软件执行后才进行评审。一旦规定了软件风险控制措施，就宜进行配置管理并评审以发现不良的副作用，包括由疏忽引发新的危险(源)或危险情况。

实施风险控制措施使得软件设计更加复杂，由此可能增加潜在的附加的软件异常或引起新的危险情况。风险控制措施宜尽可能的简单可行，并且宜始终遵从新的风险评定。

至少宜在软件设计后以及软件系统测试后，重复进行该评审。

6.7 风险控制的完整性

YY/T 0316—2016 原文

6.7 风险控制的完整性

制造商应确保所有已识别的危险情况产生的一个或多个风险已经得到考虑，这一活动的结果应记录在风险管理文档中。

用查看风险管理文档的方法检查符合性。

当软件是危险情况的影响因素时，宜考虑将 YY/T 0664—2008 的 7.3.3 导入风险控制的完整性活动中。

7 综合剩余风险的可接受性评价

YY/T 0316—2016 原文

7 综合剩余风险的可接受性评价

在所有的风险控制措施已经实施并验证后，制造商应利用风险管理计划中的准则，决定由医疗器械造成的综合剩余风险是否可接受。

注 1：综合剩余风险的评价指南见 D.7。

如果应用风险管理计划中建立的准则，判断综合剩余风险是不可接受的，制造商可以收集和评审有关资料和文献，以便决定预期用途的医疗受益是否超过综合剩余风险。如果上述证据支持医疗受益超过综合剩余风险的结论，则综合剩余风险是可接受的。否则，综合剩余风险仍然是不可接受的。

对于判断为可接受的综合剩余风险，制造商应决定哪些信息记录在随附文件中，以便公开综合剩余风险。

注 2：附录 J 提供了如何公示剩余风险的指南。

综合剩余风险的评价结果应记录在风险管理文档中。

用查看风险管理文档和随附文件的方法检查符合性。

综合剩余风险的评价要求实施所有风险控制措施。这包括软件需要在其使用的每个不同的系统配置情况下被评估。

系统测试活动(关于所有软件功能和硬件风险控制)的结果宜与可接受准则一起评价。所有剩余的软件异常需要记录在风险管理文档中，并评价以确保它们不会促成不可接受的风险(YY/T 0664—2008，5.8.2 和 5.8.3)。必要时，独立的多学科临床/应用专家的评价是可接受的。在随附文件中包括相关信息也是有必要的。

8 风险管理报告

YY/T 0316—2016 原文

8 风险管理报告

在医疗器械商业销售放行前，制造商应完成风险管理过程的评审。评审应至少确保：

——风险管理计划已被适当地实施；

——综合剩余风险是可接受的；

——已有适当方法获得相关生产和生产后信息。

上述评审的结果应作为风险管理报告予以记录，并包括在风险管理文档内。

在风险管理计划中，宜赋予具有适当权限的人员以评审的责任[3.4b)]。

用查看风险管理文档的方法检查符合性。

评审风险管理过程宜考虑 YY/T 0664—2008 的第 6 章和 7.3.3。

9 生产和生产后信息

YY/T 0316—2016 原文

9 生产和生产后信息

制造商应建立、形成文件并保持一个系统，以便收集和评审医疗器械（或类似器械）在生产和生产后阶段中的信息。

在建立收集和评审医疗器械信息的系统时，制造商尤其宜考虑：

a) 由医疗器械的操作者、使用者或负责医疗器械安装、使用和维护人员所产生信息的收集和处理机制；

b) 新的或者修订的标准。

上述系统也宜收集和评审市场上可得到的类似医疗器械的公开信息。

对于可能涉及安全的信息，应予以评价，特别是下列方面：

——是否有先前没有认识的危险（源）或危险情况出现，或

——是否由危险情况产生的一个或多个估计的风险不再是可接受的。

如果上述任何情况发生：

1) 对先前实施的风险管理活动的影响应予以评价，作为一项输入反馈到风险管理过程中；

2) 应对医疗器械的风险管理文档进行评审。如果可能有一个或多个剩余风险或其可接受性已经改变，应对先前实施的风险控制措施的影响进行评价。

评价结果应记录在风险管理文档中。

注 1：生产后监视的一些要求是某些国家法规的内容。在此情况下，可能要求附加的措施（例如未来的生产后评价）。

注 2：见 YY/T 0287—2003[8]的 8.2。

用查看风险管理文档和其他适当文件的方法检查符合性。

软件风险管理在软件生命周期内持续进行，包括软件维护过程（YY/T 0664—2008，第 6 章）和软件问题解决过程（YY/T 0664—2008，第 9 章）。

YY/T 0664—2008 的第 6 章要求制造商建立软件维护计划，阐明在医疗器械软件发布以后的接收、形成文件、评价、解决问题和跟踪反馈的程序。软件维护计划还阐明软件风险管理过程，以及软件问题解决过程，以分析和解决医疗器械软件发布后发生的问题。

软件问题解决过程（YY/T 0664—2008，第 9 章）整合了软件问题调查中和安全相关问题评价中的风险管理活动。组成一个多学科团队包括临床专家、软件工程师、系统设计人员和可用性/人因工程专家对问题调查和评价非常重要（参见 3.3）。

SOUP 也是软件维护计划和生产后风险管理活动的一个重要方面。一些 SOUP 由于其特性（如防毒软件）需要经常更新，制造商宜在软件维护计划中考虑这一点。

SOUP 的失效或非预期结果和 SOUP 软件的退市（停止技术支持）可能影响医疗器械综合剩余风险的可接受性。因此，有必要在软件系统的开发和维护中实施 SOUP 监控和评估活动。这些活动宜阐明 SOUP 的更新、升级、BUG 修复、补丁和退市。制造商应积极监控和评估关于 SOUP 实际性能的公开可得的异常列表和信息，以主动确定是否有已知的异常而导致引发危险情况的事件序列（参见 YY/T 0664—2008，6.1f），7.1.2c），7.1.3 和 7.4.2）。

制造商发布的 SOUP 软件补丁或更新可能包含附加功能，这些功能对医疗器械的安全和有效性不

是必要的。对于这些 SOUP 更新宜对多余的组件作分析，这些多余组件可以从医疗软件发布中去除从而避免可能导致危险情况的不期望的变化。

对于任何软件项的更改，制造商宜知道哪些软件项是受到 SOUP 更新影响并执行回归测试(参见 YY/T 0664—2008，7.4，8.2 和 9.7)。

附 录 A
（资料性附录）
定义的讨论

危险（源）和伤害是 YY/T 0316—2016 中一对关键的术语。对这两个术语的理解对于理解本部分很重要。伤害是对人体的损伤或对人体健康的损害，或对财产或环境的损害。危险（源）被定义为可能导致伤害的潜在根源，危险（源）有许多原因。

根据 YY/T 0316—2016 的定义，只有当事件序列或其他情况（包括正常使用）导致危险情况（参见图 A.1）时，危险（源）才能导致伤害。事件序列既包括单一事件也包括事件组合。YY/T 0316—2016 中 D.2 提供了危险（源）和危险情况的指南。YY/T 0316—2016 的附录 E 提供了关于危险（源）、可预见事件序列和危险情况示例的指南。

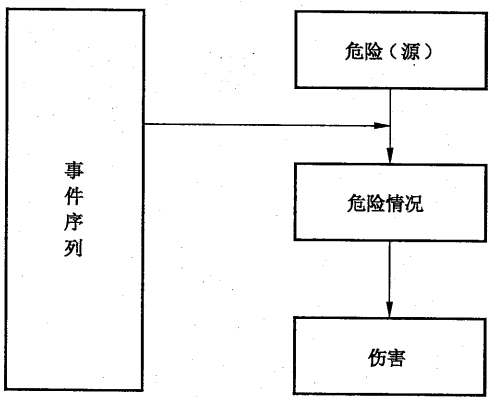


图 A.1 事件序列、伤害和危险（源）的关系

危险（源）或危险情况的起因是任何事件序列，事件的组合可以合理可预见地导致危险情况。给定的危险（源）可能有一个、几个或多个可能的原因（事件序列）。

与热能、电能或悬挂物不同，软件本身不是危险（源）（一种潜在的伤害源）；与软件接触不会造成伤害。然而，软件可能导致人体暴露在危险（源）下，换句话说，它可能促成危险情况。软件失效（任何形式）常常促进危险（源）转化为危险情况。

软件促成危险情况主要通过促成暴露危险（源）并产生危险情况的事件序列。

表 A.1 危险（源）、可预见事件序列、危险情况和可能发生的伤害的关系

危险（源）	涉及软件的可预见事件序列	初始原因	危险情况	伤害
电能	用于控制眼部植入物电流的软件输出过高	(1)软件算法有局限 (2)正确地规定了软件，但有异常	过量电流通过植入物施于患者眼部	严重灼伤 视力损伤
临床功能丧失	软件无法按照设计提供生命支持功能	(1)软件无法处理不常用的输入数据 (2)软件未检出设备的不正确设置 (3)硬件没有提供足够的资源以支持软件的及时运行	(1)设备不能在需要时提供治疗 (2)设备在不正确设置下运行 (3)设备未对威胁生命的情况做出警告	患者状况恶化 死亡

表 A.1（续）

危险(源)	涉及软件的可预见事件序列	初始原因	危险情况	伤害
临床人员对患者疏忽	软件输入和输出造成用户迷惑或误导用户	(1)软件用户界面造成用户迷惑 (2)软件输出超过用户的反应能力 (3)用户不了解软件的限制	(1)对患者的不当治疗 (2)对紧急事件缺乏及时反应 (3)对软件的过分依赖取代人的主动性	患者状况恶化 死亡

附 录 B
（资料性附录）
软件原因示例

表 B.1 列出了通常与危险（源）有关的软件功能性区域，并提供了引发危险（源）潜在原因情形的示例。同时提供了在软件开发中有益于提高风险控制的提问示例。表中的部分信息可能不适用于全部医疗器械软件，对特定医疗器械的相关性取决于该医疗器械的预期用途、系统级设计以及软件在该医疗器械中的作用和其他因素。此表只作为起始点。

此表并不详尽，但有助于考虑如何开发安全有效的软件。

表 B.1 软件功能性区域原因的示例

软件功能性区域	危险（源）原因示例	提问
报警和警示		
优先级	低优先级报警掩盖了高优先级报警的显示或可听见的输出 关键报警不持续	规范是否识别了系统如何对多报警情况做出反应？ 有多级报警吗？ 高级别报警的声音超过低级别报警吗？ 任何报警宜持续到用户确认收到报警吗？
保护性措施	每个报警情形或报警种类的预防措施不清晰 没有规定一旦报警解除就取消防护性措施	预防措施引起可用性问题吗？即用户可以通过预防措施进行安全操作吗？
关机/安全模式/恢复	安全模式措施不充分 安全模式措施引发新危险（源）	安全模式措施对于预期用途合适么？ 临床人员是否评审安全模式情景？ 安全模式状态对于用户是否明显？
用户界面	拥挤的或不良的用户界面掩盖“真实的”报警情形 响应报警的动作不清晰	临床人员评审过防护措施的可用性吗？
运行日志	持续的错误表明有未解决的故障 运行日志报警与错误的患者相关联	检出的错误记录在运行日志中了吗？ 运行日志空间够大吗？ 运行日志存储可靠吗？ 运行日志如何清除？ 运行日志清除时使用者知道吗？
可听性	背景噪声掩盖了报警声音 报警音量过大以致于操作者找到可替换的方式使报警失效 声音系统失效并且用户不知道	声音报警设计是否考虑到了预期用途的环境？ 用户界面设计的开发需求有用户参与吗？ 对于每次供电或每个患者，声音系统如何验证？
临界电源运行状态		
数据完整性	关机时非易失性的写操作正在进行中 关键参数未被保护导致开启电源重新治疗 运行时潜在的软件异常无意中修改了关键参数	电源丧失时，内存写入过程中会发生什么？ 软件知道要掉电吗？ 通电时非易失性存储验证了吗？ 在使用前检查关键参数了吗？

表 B.1（续）

软件功能性区域	危险（源）原因示例	提问
复位	意外复位后，未能同步组件 持续的复位未被检出，而这可能是即将发生故障的信号	复位作为一种风险控制措施使用吗？ 复位运行时危及输入/输出控制的安全吗？ 复位让用户知道吗？
恢复	通电初始化时医疗器械无法用于预期用途 通电时非易失性失效——你怎么做？ 用户不知道关键参数被恢复到出厂设置	恢复时间是安全问题吗？ 医疗器械可获得性是安全问题吗？ 失效的安全保护措施如何影响非易失性存储器？
电源模式	在低功率状态时，可听功能或其他关键用户界面不可用 低功率转换时，关键中断功能无意中失去	低功率模式期间，风险控制措施受影响了吗？ 软件进行验证和确认活动时，是否将从低功率模式的软件恢复作为一种可能的启动状态？
关键用户控制/可用性		
调整/导航/选择	用户界面软件过程处理数值变化，但控制过程得不到新数值	如果调整新值但没有选择或确认，用户是否被告知？ 正在被调整的参数是否需要一个两步操作以实现更改？
数据登录	用户输入了范围之外的值 用户输入了范围内但是非预期值	提示用户进行确认了吗？ 软件检查数据登录的有效性吗？ 要求监督用户登录以确认非常关键的输入或错误重写吗？
可达性	隐藏停止控制 带着外科手套时触摸屏控制不起作用	用户必须通过多少“层”导航达到安全相关功能？
屏幕变化	警告“自动地”引起屏幕显示转换	所有的自动转换屏幕都被评价了吗？
用户界面设计	使用的颜色没有考虑到常见的色盲 用户不能确定哪些情况会导致报警	色盲操作者会如何解释错误信息？ 用户界面设计的开发需求有用户参与吗？
显示		
诊断图像	方向相反 患者关联错误	有技术方法去保证正确的图像方向吗？ 图像如何与患者联系起来？
诊断波形	不合适的显示滤波器 图形失真、图像变形、缩放比例错误、时间基数错误、有损压缩	需要显示哪些频率成分？ 临床人员评审过需求了吗？ 显示滤波器的特征被完整描述了吗？即拒绝什么？通过什么？超过输入的完全范围吗？
硬件控制		
算法 电机控制	积分饱和、混叠、定时、溢出、端口错误（串行端口/并行端口）	采样率是多少？ 如果使用比例-积分-微分（PID）控制，积分增益是否受限？算法是否具备了覆盖所有硬件变化的特征？ 如果使用反馈控制，检查什么以保证反馈信号的有效性？ 微处理器和编译器使用的所有数据类型都评价了吗？

表 B.1（续）

软件功能性区域	危险（源）原因示例	提问
应用能量	没有在初始时和治疗过程中持续地检验所有的能量“阈值” 安全系统失效但是用户不知道	所有的认定是在按计划不断验证吗？ “普通模式”错误可存在于治疗控制软件和安全监控软件吗？ 每次通电或每个患者都做安全监测验证吗？
离散	位粘滞 由于轮询间隔以比特为单位的变化未被检出	软件检出位粘滞（从不变更）了吗？ 轮询速度与系统或硬件工程师讨论过了吗？
校准/自检 检验范围 分析校准 （专用校准软件）	差的使用说明书没有引导使用者对医疗器械进行正确地校准，导致错误的校准常数 在非零信号下，即非预期的袖带或线上压力，或传感器受力，进行了自动校零操作。	软件执行了合理或有效的校准值检查吗？即斜率或偏移量？ 用户知道自动校准或自动校零吗？
硬件故障检验	系统检测到硬件故障但没有报告给用户 医疗器械继续在这种条件下使用 通电后硬件故障发生 软件只对供电时硬件故障进行检查	所有的硬件故障都报告给用户了吗？ 硬件故障检查宜在通电时、每次治疗或间断的、或连续的（如每秒一次）进行吗？
自动清理	周期中，用户中断了清理或杀毒过程	软件强制该周期完成吗？ 对不完全的清理或杀毒过程的软件检出可以被破坏吗？
液体输送	检出不适当的标定 治疗过程中没有初始地和持续地检验所有的流量液体“阀门”	所有的认定是在按计划不断验证吗？ 系统安全能被破坏吗？即泵带着没有安全管夹的管子在运行。
生命支持	从未定义安全状态 在许多关断路径之外，有一条路径无法禁止中断 生命支持功能没有备份	对于目标人群（例如：成人、婴儿）范围，全面定义和分析（包括治疗延误和安全关机顺序的影响）安全状态了吗？ 软件能支持“有限功能”模式并且报告使用者情况吗？
监视		
决策	监控软件中普通模式错误 竞态条件引发错误的决策结果	治疗控制和治疗监控软件是独立开发的吗？ 对于这个决策点，软件设计是否有排除或最小化了竞态条件的可能性？
未激活	控制系统不知道监控系统已经关闭子系统 网络化系统登录时的参数未在医疗器械中被激活	控制子系统知道监控子系统的活动吗？ 未激活参数如何与用户或网络系统通信？
显示	显示值没有更新但是用户不知道 在两个或更多的优先级执行显示写操作	如何才能使用户意识到“冻结”显示？ 在被取代前，视频的“前后内容”保存了吗？
测量	错误的数据采样时间或采样率	采样率对于信号的频谱合适吗？ 测量值是贯穿所有的软件层以统一的单位储存吗？

表 B.1（续）

软件功能性区域	危险(源)原因示例	提问
接口		
不好的参数传递	函数以微升为单位进行量值传递，而驱动系统的期望值是以毫升为单位 不好的指针传递 被传递的指针指向临时的内存区域，但在处理之前内存里的值已经丢失	每个软件函数验证传递的参数了吗？ 软件语言支持更强大类型的安全检查吗？ 是不是整个软件包内使用统一的单位？ 参数是在更高优先处理层上修改吗？
网络	软件进入死循环，等待主机回应 网络上的医疗器械重“名”导致数据丢失 网络数据处理占据了中央处理机(CPU)处理周期，致使没有资源给安全或预期用途功能使用	软件被设计成默认各种网络连接环境吗？ 远程连接能通过重复地发命令或假数据降低系统性能吗？ 医疗器械检查网络名是否已被使用了吗？
数据		
临床信息	系统访问错误患者记录，而用户界面显示没能使其显而易见 系统以错误的存档方式储存患者数据	有多样独立的标识符显示，使用户循环检出混乱吗？ 关键标识符可以嵌入真实数据用作反复核对吗？
报告	报告提供错误的数据或以错误的顺序识别或没有单位	什么报告会被用于临床？ 如果数据不正确伤害的严重度有多大？ 临床医生有多少可能注意到这个问题？
数据库	由于系统级失效的副作用或未知来源软件(SOUP)副作用导致数据崩溃	怎样在使用前发现数据崩溃？ 可以在每次使用时检查而不只是在启动时吗？
诊断		
决策	伪影侦测指示抑制了显示器上的心脏停跳指示	警报指示的层级得到充分评审，并与临床人员进行了评审吗？
数据转换	运算精度错误导致无效结果 运算使用或显示了错误的单位	需要什么样的运算精度？ 宜如何进行数学公式编码以保证充分的精度？
自动的预防性维护	后台诊断临时更改数据但是与此同时应用代码正在检索实际使用的数据 后台诊断干扰了正常的时序	诊断中，应用过程在适当的时间被锁定了吗？ 关键的定时周期内诊断被锁定了吗？
保密安全		
配置选项	对访问关键配置参数或数据没有保护或保护不充分	什么数据是关键的，用户不宜更改或者只有在监督授权时才可以更改？ 需要跟踪监控吗？
功能性访问	对访问治疗控制或仪器运行没有保护或保护不充分	操作前操作者需要登录吗？ 患者能不经意的操作医疗器械吗？
接口访问	对通过通信接口或网络提交的数据或指令没有保护或保护不充分	什么是被允许远程操作的？ 远程系统设想的控制可靠吗？如果可靠，为什么？

表 B.1（续）

软件功能性区域	危险（源）原因示例	提问
性能		
容量/负载/响应时间	峰值负载时关键时序受到影响 处理/输入/输出的顺序受到影响或峰值负载下数据丢失 在系统峰值负载下电机控制受到影响	峰值负载或当达到容量限值时，数据或时序会在未察觉的情况下丢失或受影响吗？ 峰值负载下，输入和输出会在正确的顺序下排队等候吗？ 关键的功能和风险控制措施在这些条件下测试了吗？ 是否实施了风险控制措施以探测限值？ 能否设置中断将受临界时间约束的功能与其他功能隔离？

除了表 B.1 显示的潜在软件原因导致的安全相关软件功能危险（源），还有一些类型的软件原因可导致软件副作用，而这与产生失效的软件无关。如果某种软件缺陷对软件的安全相关部分有不可预期的影响，就宜识别这种潜在性，并开发风险控制策略及具体的风险控制措施。

表 B.2 列举了这些危险（源）的潜在软件原因示例和每个示例的可能风险控制措施。基于需求的系统测试，在识别这些软件原因或者验证相关的风险控制措施及风险控制措施的有效性方面常常是无效的。表中的右列提供了适用于这些示例的相应类型的静态或动态的验证方法的指南。表 B.3 提供了静态或动态验证方法的示例。

表 B.2　带来副作用的软件原因示例

	验证类型	分析：静态（static）/动态（dynamic）/时序（timing）		
		测试（单元、集成）		
		检查		
软件原因	风险控制措施			
算法				
除以零	运行时间错误俘获，防备性编码	◆	◆	D
数字上溢/下溢	范围检查，浮点数表示	◆	◆	D
浮点数取整	鲁棒算法	◆		
不合适的范围/边界检查	防备性编码	◆	◆	S
差一错误（OBO）	防备性编码	◆	◆	
硬件相关				
电可擦只读存储器（EEPROM）用法： 长时间访问，疲劳	使用特殊的访问模式（页面模式/突发模式），只在数据变更时写操作，高速缓冲存储器写操作，只在失电时更新电可擦只读存储器（EEPROM）	◆		T
中央处理机（CPU）/硬件失效	开机中央处理机（CPU）检查，映像程序循环冗余码校验（CRC）检查，随机存取存储器（RAM）测试，时钟检查，看门狗检查，非易失存储器检查，对硬件响应的超时和合理性原因检查，传感器与电源/接地短路检查，用已知信号测试传感器响应		◆	

表 B.2（续）

	验证类型	分析：静态（static）/动态（dynamic）/时序（timing）		
		测试（单元、集成）		
		检查		
软件原因	风险控制措施			
噪声	防抖数字输入，滤波器模拟输入，全部中断—使用的和未使用的一都有中断服务程序（ISRs）			
外围接口异常	模数转换器（ADC）/数模转换器（DAC）导致启动延迟，验证时序和其他接口需求总是匹配，合理性检查	◆		T
时序				
竞争条件	更新时识别和保护（锁定）共享资源，执行单一的、不能重新进入的过程来处理对共享资源、共享资源分析的所有访问			S
错过时间最后时限	特定的时序需求，适当的时序设计算法，通过设计排除优先反转和死锁问题，避免非确定的时序构造，在完整代码中验证所有的时序假定 注：非确定时序结构包括：回归程序，等待硬件响应，动态内存分配和虚拟内存（与硬盘交换存储页）			T
错过中断功能	简单中断结构（最少的优先级），中断服务程序（ISRs）简短快速，软件中断阻塞是罕见的并且持续时间短，适当的实时设计			T
输出抖动过大	中断服务程序（ISRs）简短且执行快速，避免非确定的定时构造，适当的实时设计，在周期任务开始和高优先级的中断服务程序（ISRs）时更新所有输出			T
看门狗超时	适当决定看门狗计时器重置和设置暂停时间之间的最长时间间隔，避免危险情况的同时还要尽可能的设置最长周期的暂停时间			T
模式				
异常终止	退出俘获，运行错误俘获，适当看门狗计时器设计，程序输入的有效性检查，供电自检，发布前删除程序中的“调试”代码和其余非产品功能，确保“特殊”模式（服务，制造）不能轻易进入			D
电源丧失/恢复/排序问题	启动检查：中央处理机（CPU）、映像程序循环冗余码校验（CRC）、随机存取存储器（RAM）、时钟、看门狗、非易失存储器、外围设备等。适当状态设计，时间平均值初始化，外围设备重新初始化，从非易失性存储库中储存/恢复系统，外部电压监控/重置电路			
启动/关闭异常	启动检查（参见上方），外围设备和数据适当初始化，适当时使用非易失性内存，适当的状态设计			
进入/退出低电量模式	适当的中断设计	◆		T

表 B.2（续）

	验证类型	分析：静态（static）/动态（dynamic）/时序（timing）			
		测试（单元、集成）			
		检查			
软件原因	风险控制措施				
数据问题					
数据崩溃	映像随机存取存储器（RAM）、循环冗余码校验（CRCs）或校验和模块，通过函数封装数据访问，全局数据减到最少，保持数据结构简单，知道编译器如何在结构中排列数据，避免类型转换（参见下方的“错误指针”和“中间数据”）				S
资源竞争问题	共享资源分析（参见上面“竞态条件”）				
错误指针	防错性编码：取消定位前做有效性测试，使用强类型语言，将指针使用减到最少，避免指针转换		◆	◆	S
数据转换错误：数据类型转换，缩放	避免类型转换，转换使用浮点数表示		◆	◆	S
初始化错误	预设置时均变量，加电时所有数据存储器清零		◆	◆	S
范围外的平均数据	确保计算均值时有足够的采样（尤其是上电时），或者将均值预先初始化为已知（或最后一个）好的数值		◆		
转滚法	合理性检查		◆	◆	
易失性数据	验证易失性存储器类别被用来表达共享数据，这些数据是被硬件、ISRs 或不同优先级任务所共享的		◆		S
图形失真	采样数据的频率至少要大于被采样信号最高频率成分的频率的两倍（奈奎斯特准则），限制信号的带宽				
使用中间数据 注：当运算可以被中断或优先占用时，不要执行针对全局变量（或共享）的一系列计算，改为使用暂时变量运行所有的计算和用单一的、不可中断的指令更新全局变量	确保被假设要及时同步的任意数据立刻得到全部更新，共享资源分析		◆		
界面问题					
更新显示失效	持续更新而不是事件驱动				
人为因素：误用	重建情形的日志，上下文相关的帮助，简单的用户界面设计				
网络问题，如：多用户	负荷测试				T
硬件/软件配置错误/驱动程序错误	软件开发流程，配置管理工具				
破损的补丁/更新	启动时映像程序循环冗余码校验（CRC）和版本检查，检查协议修订，产品有效期				

表 B.2（续）

	验证类型	分析：静态（static）/动态（dynamic）/时序（timing）		
		测试（单元、集成）		
		检查		
软件原因	风险控制措施			
未知来源软件（SOUP）失效模式：中止/不回复，锁定中断时间过长	检查 SOUP 勘误表，鲁棒设计（阻塞调用超时检查），锁定经常使用或被 ISRs 使用的内存页，使用只有 SOUP 特性所必需的：除去其他	◆		T
病毒	病毒检查器			
不兼容浏览器/网络	启动时集成版本检查、兼容性测试			
杂项				
内存溢出	避免动态内存分配	◆	◆	D
系统死锁	简单锁策略（在给定的时间内进程只能有一个资源锁住），死锁分析			S
重入	确保所有被中断所调用的（或不同优先级的多任务）功能，包括那些包含在那些第三方库里的，能再次执行			D
堆栈溢出	堆栈执行时期保护，高水位标识，堆栈分析			S
逻辑错误/语法	使用资源代码分析工具（如 Lint）和/或最大值。编译器警告级别在临界控制点进行双重多样化和相互校验	◆	◆	S
代码损坏	启动和运行时映像程序循环冗余码校验（CRC）检查			
无作用程序代码	如未移除无作用程序代码而插入纠错，在无作用程序代码（在定制的或成品软件的组件中）运行后会警示或执行安全关机			D
错误条件代码	确保适当时和仅当需要时使用条件编辑	◆		
非预期的宏副作用	在所有宏参数处使用括弧			S
资源损耗	堆栈，堆和时序分析			T
错误警告/报警优先次序	压力测试			
未授权的要素（“镀金”“后门”等）	需求和设计评审，跟踪矩阵			
操作/优先顺序错误	“Bread-crumbing”跟踪呼叫			
安全状态	独立监视器			

有许多方法有利于保证风险控制措施按预期执行，其中有些需要更密集的资源。没有充分的单一方法。见表 B.3 中识别的一些方法。

表 B.3　有利于保证风险控制措施按预期进行的方法

静态分析	动态测试	建模
走查	功能测试	环境建模
设计评审	时序和内存测试	时序仿真
寄生电路分析	边界值分析	用例/用户工作流
	性能测试	
	压力测试	
	统计测试	
	错误猜测	
	基于线程的测试	
	基于使用的测试	
	群集测试	

测试宜考虑多种类型(例如,压力、边界、时序、失电、故障、SOUP 失效等)以保证安全相关软件在足够大范围的条件下测试,而不是只关注于基于需求的测试。

附 录 C
（资料性附录）
软件相关的潜在隐患

表 C.1 列出了需要在风险管理活动（针对 YY/T 0316—2016 的条）和软件生命周期（针对 YY/T 0664—2008 的条）中避免的软件相关的潜在隐患。

表 C.1 需要避免的软件相关的潜在隐患

YY/T 0316—2016 的第 4 章：风险分析
——使用不切实际的低概率估计软件失效，会产生不真实的风险等级，导致不适当的风险控制措施； ——增加了软件特性，而没有进行风险分析来确定是否有新的危险（源）和危险情况或原因被引入该医疗器械，或者没有进行风险分析来确定现有的风险控制措施是否受到弱化（可以是在软件初始开发阶段也可以软件发布后作为维护的部分）； ——医疗器械风险分析过程仅定义了系统和硬件层面，既没有充分的阐述软件与风险分析的关系，也没有要求特殊考虑软件异常作为危险（源）和危险情况的潜在原因； ——风险分析和软件开发生命周期程序的严密性与医疗器械的潜在伤害不相适应
YY/T 0316—2016 的 4.1 风险分析过程
——风险分析过程只定义了系统和硬件层面，软件仅在对硬件失效实施风险控制措施时被提及； ——风险分析和软件开发生命周期程序的严格程度与医疗器械的潜在伤害不相适应； ——软件仅在产品开发生命周期的末期被视为风险分析的一部分
YY/T 0316—2016 的 4.2 预期用途识别
——仅考虑用户环境的一个子集/仅考虑潜在的计算机系统平台的一个子集； ——不考虑平台演变或对保密安全的需要或其他未知来源软件（SOUP）补丁程序； ——对误用和使用错误引发的潜在危险（源）考虑不充分，因而与其相应的风险控制措施没有被识别
YY/T 0316—2016 的 4.3 危险（源）的识别
——仅使用失效模式和效应分析（FMEA）或者故障树分析（FTA）方法，就以为单独一个用于风险管理是充分的； ——对硬件和软件孤立的使用失效模式和效应分析（FMEA）或者故障树分析（FTA）； ——忽略整体的危险（源）级别及原因如： ● 有不可预见影响的软件错误； ● 用于硬件失效风险控制措施的软件逻辑错误； ● 医疗器械的预期临床用途的软件逻辑错误（如结果计算的算法）； ● 软件平台失效-操作系统、库、SOUP； ● 计算机组件和外围设备失效； ● 通信接口失效； ● 人为因素。 ——通过下面的假设进行原因识别： ● 软件异常仅仅影响特定组件的功能，对其他的软件项或数据没有副作用； ● 软件会正确地运行； ● 潜在的软件失效原因过多，并且在识别、检测或风险控制上不可预测； ● 在导致危险情况的软件事件序列的初始或结束使用风险控制措施就足够

表 C.1（续）

YY/T 0316—2016 的 4.4　风险估计
——假设单一故障条件的概念适用于软件设计的问题和事件序列； ——假设不详尽的测试将某个失效的概率降低至零； ——基于功能，假设某些软件项不是安全相关的，而不考虑潜在的非预期的副作用； ——在没有足够临床知识情况下，或者在没有人为因素相关的临床知识[涉及危险（源）对所有潜在用户和目标人群的影响]的人参与的情况下，就赋值严重度； ——基于临床医生会发觉失效或错误信息的假设，而赋值低的严重度； ——基于所有用户都正确的遵循器械标签和手册或者所有用户不会因疏忽造成错误的假设，而赋值低的严重度； ——假设某个危险（源）已计划了风险控制措施，并作为初始赋值严重度的一部分考虑。如果此假设有误，低初始严重度可能导致在以后识别的风险控制不充分； ——仅使用潜在的对患者的直接伤害来判定严重度，而不考虑通过软件提供给用户信息的间接使用，不考虑延迟治疗和其他与医疗器械有效性和基本性能有关的因素； ——假设临床医生总会反复核查软件提供的信息，或会发觉错误信息而赋值低严重度，基于此而不实施其他的风险控制措施
YY/T 0316—2016 的第 5 章　风险评价
——对软件异常可能性的主观判断，以确定其不需要风险控制措施； ——因硬件特征而从软件方面考虑消除危险（源），后来硬件修改或者被移除导致在某种程度上使软件成为可能的影响因素，但却未考虑额外的软件风险控制措施； ——由于假设软件会按预期运行或者假设通过测试会发现所有异常，而没有考虑潜在的软件异常是导致危险（源）的一个因素
YY/T 0316—2016 的 6.3　风险控制措施的实施
——风险控制措施在正常条件或限定条件下被验证，而不是在大范围异常和压力条件下被验证； ——用于完成风险控制措施的软件或数据处于组件中或其他软件易于访问到的位置中，使潜在的危险（源）的副作用增大； ——风险控制措施只在一个操作平台或程序变量上被验证； ——一些风险控制措施因为很难强制发生风险（例如内存失效、竞争条件、数据崩溃、堆栈溢出）而没有被实际验证； ——假设在开发过程能发现所有安全相关异常，测试会保证其在现场正常工作； ——风险控制措施的执行使软件设计明显更复杂，这种复杂性增加了额外软件异常的可能性或引发新危险（源）
YY/T 0316—2016 的第 9 章　生产后信息
——当引入另外的风险控制措施时，忽略了因使用错误而产生潜在危险（源）的现场事件； ——没有现场信息评价就假设初始概率和严重度估计是正确的； ——没有考虑到医疗器械用于非预期的用途，此时已执行的风险控制措施可能不充分。例如用于测试艾滋病病毒（HIV）的体外诊断试剂预期用于个人用途，但却被用于筛查公众的血液供应
YY/T 0664—2008 的 5.1　软件开发策划
——在软件计划和生命周期过程中没有建立风险管理活动； ——软件风险管理活动和整体的医疗器械风险管理活动不相关联； ——软件风险评定只在生命周期的一个阶段进行； ——软件开发者和测试者未经培训或没有风险管理的经验； ——假设常规的风险管理活动覆盖了软件风险管理； ——软件风险并没有规范化管理； ——安全决策的可追溯性没有确立

表 C.1（续）

YY/T 0664—2008 未知来源软件(SOUP)考虑
——由于定义软件结构时没有考虑风险评定和风险控制，漏掉固有安全设计的风险控制措施； ——假设测试会使无效的体系结构足够安全； ——未能识别体系结构的安全相关方面，当这些结构要素随后被改变或删除时，会导致未知的安全风险
YY/T 0664—2008 的 5.4　软件详细设计
——只关注正常情形的处理，且假设组件间的接口和参数的传递将会是正确的，而不是纳入多重等级的错误检查； ——对详细设计的头脑风暴和后续评审中，没有考虑到识别能导致危险（源）和危险情况的潜在软件失效及相关的风险控制措施； ——在风险管理活动中忽略软件失效原因（参见附录 B）
YY/T 0664—2008 的 5.5　软件单元的实现和验证
——相信最好的编码和/或测试过程、惯例、工具或雇员，能弥补差的、固有不安全或过于复杂的设计； ——在关键的代码设计中使用缺乏经验的开发者； ——未能定义和要求明确的防御性编程惯例； ——仅仅依赖动态测试而不执行代码检查或静态代码分析，尤其是对关键组件； ——在不理解风险管理和设计需求的相关性的情况下而偏离设计； ——在开发过程早期仅在关键组件上运行一次单元测试，而不把其当作回归测试的一部分重复测试； ——将测试仅集中于动态、黑盒、系统层面的技术而不执行静态和动态的白盒验证
YY/T 0664—2008 的 5.6、5.7 软件集成、集成测试和系统测试
——不使用风险评定信息来策划测试或培训测试者； ——尽管百分之百的测试是不可能做到的，但还是靠测试作为风险控制措施； ——没有把系统或软件故障模式的仿真作为测试的一部分来验证风险控制措施； ——使用没有资质的和不受控的自动测试工具并且信任结果； ——未能正确的分析代码来查明测试中不能发现的异常
YY/T 0664—2008 的 5.8　软件发布
——未能使已发布的软件和文档版本保持一致，以致对开发和测试团队未来产品的发布造成误导。不准确的文档和不准确的可追溯性能够导致失去联系，能够导致忽略危险（源）及其原因，失去风险控制措施或未能充分的验证安全相关软件； ——未能使有足够临床经验的人参与到剩余异常的评价中； ——剩余异常的重要性的评价仅仅基于检测到的功能特性而非完整的根本原因分析，就来判定某些条件下的所有潜在的副作用； ——忽略一旦第三方不再发布特定的 SOUP 版本，这些版本对于失效调查和现场纠正将不可获得这一事实； ——因特定工具及其版本（如编译器）没有存档，失去创建特定软件版本的能力； ——医疗器械的寿命可能会比当前的存档媒介长，当制造商使用新的存档媒介取代旧的存档媒介时，宜策划一个将旧的存档转移到新媒介的途径
YY/T 0664—2008 的 6.1　制定软件维护计划
——制定维护过程时，对于变更的风险管理没有清晰的方法； ——制定变更的风险管理时，仅描述变更的预期功能，而没有涉及受影响的组件和与其相关的风险

表 C.1（续）

YY/T 0664—2008 的 6.2-6.3　问题和修改的分析和实施
——假设一个小的功能变化不会影响安全； ——在没有重新检测现有的风险控制措施和用户界面适当性的情况下，将医疗器械的用途扩展至新的目标人群、新的病征、新的用户类型（如护士而非外科医生）或新的平台； ——没有确定根本原因和潜在的副作用，基于已报告的现场问题的症状而设定问题解决资源的优先级； ——为非临床目的（如开账单）而设计的软件包含了临床数据，这些临床数据后续用于临床目的，这种情况没有适当的风险管理

附 录 D
（资料性附录）
生命周期/风险管理矩阵

表 D.1 列出了 YY/T 0664—2008 的开发活动和相关的软件风险管理活动。注意该表格并不旨在展现严格的按次序排列的瀑布型生命周期。

表 D.1 生命周期/风险管理关系表

YY/T 0664—2008 生命周期要求	YY/T 0316—2016		
	4 风险分析	5 风险评价	6 风险控制
5 软件开发过程			
5.1 软件开发策划	风险管理策划（YY/T 0316—2016 的 3.4） 计划和文档： a) 策划的风险管理活动范围，识别和描述医疗器械和适用于计划每个要素的生命周期阶段 b) 职责和权限的分配 c) 风险管理活动的评审要求 d) 基于制造商确定可接受风险方针的风险可接受性准则，包括在伤害发生概率不能估计时的可接受风险的准则 e) 验证活动 f) 关于相关的生产和生产后信息的收集和评审活动		
5.2 软件需求分析	● 分析预期用途、合理可预见的误用和用户以识别危险（源） ● 识别已知的和可预见的危险（源）与临床医生、患者、服务人员及其他任何接触该医疗器械的人的关系 ● 考虑产品阶段：如安装和装配、培训、使用、升级和维护 ● 识别能够导致危险情况的事件的序列和组合 ● 估计每个已识别的危险情况的风险，考虑可能结果的严重度 ● 4.3 软件安全分类（YY/T 0664—2008 要求）	● 对已识别的风险，确定是否需要降低风险 ● 确定是否仅软件风险控制措施就足够了，或硬件风险控制措施是否是必要的或期望的和可行的	● 为已识别的风险（如因硬件失效和使用错误造成）识别软件风险控制措施 ● 对可能影响设计（如固有安全设计或防护措施）的软件失效进行软件风险控制措施的初始识别 ● 识别软件以提高危险情况的可检测度并降低其严重度和/或降低其发生概率 ● 对新危险（源）评审风险控制措施

表 D.1（续）

YY/T 0664—2008 生命周期要求	YY/T 0316—2016		
	4 风险分析	5 风险评价	6 风险控制
5.3 软件结构设计	● 识别关键数据与组件以及可导致危险（源）的缺陷级别，特别需要注意软件原因 ● 识别相关危险（源） ● 识别接口：通信内容是什么和何时进行通信 ● 评价性能标准和限值 ● 4.3 软件安全分类（YY/T 0664—2008 要求）	● 从风险的角度重新评价软件被赋予的角色的可接受性 ● 评价受非安全相关功能影响的控制措施的敏感性	● 识别结构风险控制措施来隔离关键组件，预防或检出危险（源）的特定软件原因 ● 特别注意要提供适当的冗余 ● 识别和冗余相关的危险（源） ● 识别检出和控制的总体方法
5.4 软件详细设计	● 识别危险（源）的其他潜在原因 ● 假设数据、编码和传输错误 ● 假设硬件失效	● 重新评价风险控制措施的充分性 ● 确定区分安全相关的代码和非安全相关的代码	● 执行特定的风险控制措施和防御性设计/编程惯例 ● 完成可追溯性和覆盖率分析来确保风险控制措施得以实施 ● 确定是否实现了未规定的功能
5.5 软件单元的实现和验证	● 识别危险（源）的其他潜在原因 ● 执行同类代码的测试故障评价	● 通过质疑一系列条件下的风险控制措施和在有代表性的环境下选用有代表性的用户，来测试重新评价风险控制措施的充分性	● 在一系列条件下的平台上验证风险控制措施 ● 最终发布之前进行风险控制措施的回归测试 ● 完成可追溯性和覆盖率分析以确保风险控制措施已执行和已测试
5.6 软件集成和集成测试			● 最终发布之前进行风险控制措施的回归测试 ● 完成可追溯性和覆盖率分析以确保风险控制措施已执行和已测试
5.7 软件系统测试			● 最终发布之前进行风险控制措施的回归测试 ● 完成可追溯性和覆盖率分析以确保风险控制措施已执行和已测试
5.8 软件发布	● 识别配置管理计划，包括配置项和依存关系		● 验证定制软件和 SOUP 的正确版本已发布 ● 验证构建的环境处于配置控制之下

表 D.1（续）

YY/T 0664—2008 生命周期要求	YY/T 0316—2016		
	4 风险分析	5 风险评价	6 风险控制
6 维护过程			
6.1 制定软件维护计划	● 策划对于变更、增强和安装如何进行风险管理，以及如何监控和分析现场的使用信息，以评估风险控制的充分性和额外的风险降低机会		
6.2 问题和修改分析	● 分析现场性能以识别先前未认识到的或其他的危险（源）及这些危险（源）的原因	● 重新评价风险控制措施的风险评级和充分性	● 识别是否需要额外的风险控制措施或是否有必要修正已有的措施
6.3 修改的实施	● 与开发过程相似但要关注变化对以下的影响： ● 影响已有的风险控制措施 ● 引入新的导致危险（源）的原因 ● 引入新的预期用途功能，以至引入新的危险（源） ● 安全相关代码的回归测试 ● 软件发布依照 YY/T 0664—2008 的 5.8		

附 录 E
（资料性附录）
安全用例

安全用例是“一种通过提供令人信服的、易于理解的、有效的案例等大量证据支持的结构性的论点，用于表明医疗器械在给定的操作环境里对于给定的预期用途是安全的。”（改编自 UK MoD Def Stan 00-56）。

在如军事系统、近海石油、铁路运输和原子能等行业，安全用例的概念是被广泛认可的，这种技术对于医疗器械行业不是强制的，这个附录也不是为了发起超出 YY/T 0316—2016 之外的要求。

本部分提出的安全用例是一种结构化、文件化和沟通方法以证明医疗器械的安全达到充分的水平。安全用例也能有助于确保在整个医疗器械生命周期内安全得到保持。

安全用例使用风险管理过程的结果来说明为什么软件对于其预期用途足够安全，以及为什么软件能满足所有的法规要求（同样适用于相关法规术语）。

可以把安全用例看成风险管理或剩余风险总结，该总结引用更详细的文档支持信息及风险管理文档中的证据。安全用例也能包含交叉引用以证明规范和对所有的风险控制措施测试覆盖率。

为实施安全用例，需要以下步骤：

——明确系统的一系列要求；

——提供支持的证据；

——安全论点的集合，将要求和证据联系起来；

——支持论点的假设和判断；

——允许不同的观点和细节水平。

安全用例的主要要素：

——要求：关于系统或一些子系统的特性；

——证据：用作安全论点的基础。可以是事实（例如基于已经建立的科学原理和先前的研究）、假设或子要求，从低一层的子论点中产生；

——论点：将证据和要求联系起来，可以是确定性的、可能性的或定性的；

——推论：提供论点转化规则的机制。

关于安全用例更多的要素和构造的信息，参见[9]。

提供安全用例介绍和结构化目标注释的两篇文章是[10]和[11]。

参 考 文 献

注：联合工作组并未认可下列的任何科技参考书的内容，这些书目是以 ISO 14971 对医疗器械软件要求的应用指南的相关附加信息的方式提供的。

[1] ISO 13485, Medical devices—Quality management systems-Requirements for regulatory purposes

[2] IEC 60812, Analysis techniques for system reliability-Procedure for failure mode and effects analysis (FMEA)

[3] IEC 61025, Fault tree analysis (FTA)

[4] IEC 61882, Hazard and operability studies (HAZOP studies)-Application guide

[5] IEC 62366, Medical devices-Application of usability engineering to medical devices

[6] IEC 80001-1, Application of risk management to information technology (IT) networks incorporating medical devices—Part 1: Roles, responsibilities and activities

[7] PULLUM, L. Software fault tolerant techniques and implementation. Boston: Artech House, 2001

[8] BANATRE, M., LEE, P. (Eds)., Hardware and Software Architectures for Fault Tolerance: Experiences and Perspectives. Berlin, Germany: Springer-Verlag, 1994

[9] BISHOP, P., BLOOMFIELD, R. (1998), A Methodology for Safety Case Development, Safety-Critical Systems Symposium http://www.adelard.co.uk/resources/papers/pdf/sss98web.pdf.

[10] KELLY, T. P., Systematic Approach to Safety Case Management, Proceedings of SAE 2004 World Congress, Detroit, March 2004 (Proceedings published by the Society for Automotive Engineers).

[11] WEAVER, R. A., KELLY, T. P., The Goal Structuring Notation-A Safety Argument Notation, Proceedings of the Dependable Systems and Networks 2004 Workshop on Assurance Cases, July 2004.

定义术语的索引

YY/T 1406.1—2016/IEC/TR 80002-1:2009

YY/T 1406.1-2016

*
书号:155066·2-31114
定价: 48.00 元

打印日期:2017年3月3日 F007