# 故障树 FTA

## 分析方法

### 概念

故障树分析是一种基于故障的演绎方法。

作为一种演绎方法，故障树分析从一个不希望发生的事件开始，例如主机故障，然后用一个系统的、向后的过程来确定（推断）其原因。在确定原因时，故障树（FT）被构造为事件及其关系的逻辑图示，这些事件和关系是导致意外事件或顶层事件的必要和充分的。

故障树中使用的符号表示所涉及的事件类型和关系类型。

故障树是一个定性模型，它提供了关于意外事件原因的非常有用的信息。

故障树也可以被量化，以提供关于顶层事件发生的概率以及FT中建模的所有原因和事件的重要性的有用信息。

除故障树分析外，归纳法也可用于安全分析、风险和可靠性分析。与FTA中使用的演绎方法不同，归纳法是一种从基本原因或始发事件开始，然后调查（归纳）最终结果的前进式方法。故障树分析和归纳法都是基于故障的。

一个FT可以转换成它的逻辑补码，一个成功树（ST），它显示了防止意外事件发生的具体方法。ST提供的条件是，如果得到保证，可以保证不发生意外事件。ST是一个有价值的工具，它提供了与故障树相同的信息，但从成功的角度来看。描述了将FT转换为其ST的技术以及ST的用途。

FTA提供了关键信息，可用于优先考虑非期望事件的参与者的重要性。FTA提供的贡献者重要性生动地显示了占主导地位的原因，这应该是任何安全或可靠性活动的重点。更正式的风险收益方法也可以用来优化资源分配，以最小化资源支出和意外事件发生的概率。这些风险效益方法对于分配资源支出非常有用，例如对航天飞机这样的复杂系统进行安全升级。

故障树分析可以应用于现有系统和正在设计的系统。当FTA应用于不存在特定数据的系统时，FTA可以使用通用数据对设计组件或概念进行分类，从而提供失效概率和重要因素的估计值。FTA也可以作为基于性能的设计开发中的一个重要元素。当应用于现有系统时，FTA可用于识别弱点和评估可能的升级。它也可以用来监测和预测行为。此外，故障树分析可以用来诊断原因和潜在的纠正措施，观察到的系统故障。获取这些信息的方法和工具以及这些信息在决策中的应用是AFTH的重要课题。

AFTH的第二部分包含了FTA在先前研究中的应用实例。重点是航空航天应用。这些例子包括压力箱破裂（典型的故障树分析例子）、单组元推进系统中无法启动和终止推力、冗余容器密封失效（设计分析）以及飞行任务航空电子系统的动态FT分析。

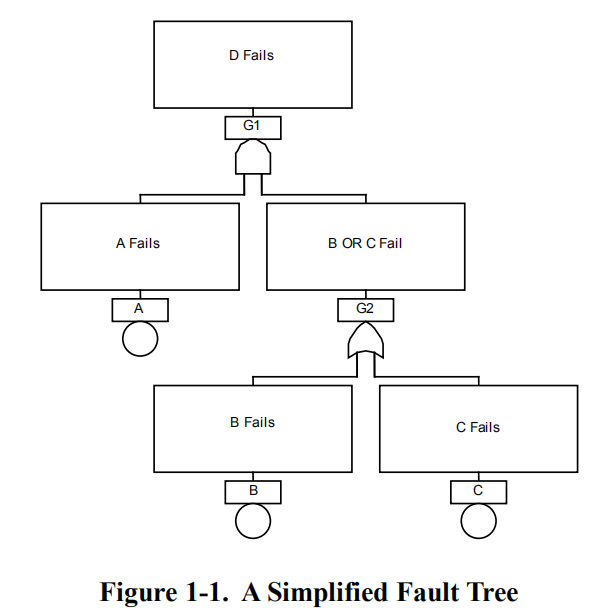
### 故障树方法

故障树分析可以简单地描述为一种分析技术，通过这种技术，可以指定系统的非期望状态（通常是从安全或可靠性的角度来看是关键的状态），然后在其环境和操作的背景下对系统进行分析，以找出可能发生意外事件（顶层事件）的所有现实方式。故障树本身是各种并行和顺序故障组合的图形模型，这些故障将导致预定义的意外事件的发生。故障可以是与组件硬件故障、人为错误、软件错误或可能导致意外事件的任何其他相关事件相关的事件。因此，故障树描述了导致意外事件（故障树顶部事件）的基本事件的逻辑相互关系。

重要的是要明白，故障树不是所有可能的系统故障或系统故障的所有可能原因的模型。故障树是根据其对应于某个特定系统故障模式的顶层事件而定制的，因此故障树仅包括导致该顶层事件的那些故障。此外，这些故障并非详尽无遗，它们仅涵盖分析员评估为现实的故障。

还需要指出的是，故障树本身并不是一个定量模型。它是一种可以定量评估的定性模型，通常是这样。当然，这种定性的方面实际上适用于所有类型的系统模型。事实上，故障树是一个特别方便量化的模型，这一事实并不会改变模型本身的定性性质

故障树固有的概念是，结果是一个二进制事件，即成功或失败。故障树由一个称为“门”的复杂实体组成，这些实体用于允许或禁止故障逻辑向上传递。门显示了“更高”事件发生所需的事件之间的关系。“高”事件是门的输出；“低”事件是门的“输入”。门符号表示输出事件所需的输入事件的关系类型。图1-1显示了一个简单的故障树



### 故障树的定性和定量评价

定性和定量评估都可以在FT上进行。FT本身是对导致顶级事件的事件和关系的定性评估。在构建故障树的过程中，我们获得了关于顶级事件原因的重要见解和理解。其他评估有助于进一步完善故障树提供的信息。

定性评估基本上将FT逻辑转换为逻辑等价形式，从而提供更集中的信息。得到的主要定性结果是顶事件的最小割集minimal cut sets（MCSs）。割集是可以导致顶部事件的基本事件的组合。MCS是产生顶级事件的最小基本事件组合。基本事件是故障树的底部事件。因此，最小割集直接将顶层事件与基本事件原因联系起来。顶级事件的MCS集合代表了基本事件导致顶级事件的所有方式。最小割集的一个更具描述性的名称可能是“最小故障集”。MCS集不仅可以用于顶部事件，还可以用于FT中的任何中间事件（例如，门事件）。

从MCSs的结构可以得到大量的信息。具有一个基本事件的任何mc标识一个单独的故障或单个事件，这些故障或事件本身就可能导致顶层事件的发生。这些单一故障往往是薄弱环节，是升级和预防措施的重点。此类单一故障的例子包括可能导致系统故障的单个人为错误或单个组件故障。具有相同特征的事件的mc表示对隐式相关故障或共同原因的敏感度，这可能会否定冗余。一个例子是相同阀门故障的MCS。单个制造缺陷或单一环境敏感性可导致所有阀门同时失效。

FT的定量评估包括**顶事件概率**和**基本事件重要性**的确定。任何量化结果的不确定性也可以确定。故障树通常通过**计算每个最小割集的概率**和**求和所有割集概率**来量化。然后根据概率对割集进行排序。对顶事件概率有显著贡献的割集称为**优势割集**。虽然顶层事件的概率是分析的主要焦点，但也可以确定故障树中任何中间事件的概率。不同的应用可以计算不同类型的概率。除了通常计算的恒定概率值外，还可以计算与时间相关的概率，以提供顶部事件首次出现的时间的概率分布。还可以计算最高事件频率、故障或发生率以及可用性。如果顶层事件是系统故障，这些特征尤其适用。

除了主割集的识别外，FT中事件的重要性是可以从FT量化中获得的一些最有用的信息。量化的重要性允许根据引起顶级事件的事件的重要性对行动和资源进行优先级排序。基本事件、中间事件和最小割集的重要性可以确定。对于不同的应用，可以计算不同的重要性度量。一个衡量标准是**每个事件对最高事件概率的贡献**。第二个是如果事件被阻止发生，最高事件概率的降低。第三个衡量标准是，如果保证事件发生，最高事件发生概率的增加。

这些重要性度量用于确定优先级、预防活动、升级活动以及维护和维修活动。后面的章节将进一步详细描述从FT中可以获得的丰富的定性和定量信息

### 成功树作为故障树的逻辑补充

因为成功和失败是相关的，所以可以将FT转化为等价的ST。在FT上下文中，成功树中的成功被明确定义为未发生的顶层事件。从FT获得ST的方法将在后面的章节中描述。这个

ST是FT的逻辑补码，ST的top event是FT的top event的补码。例如，如果FT的top event是“LOV的发生”，LOV意味着车辆丢失，那么ST将有一个top event“未发生LOV”

因此，ST定义了故障顶部事件未发生的逻辑。此外，ST确定了需要预防的最小基本事件集，以确保不会发生故障顶部事件。这些防止故障顶部事件的最小事件集称为最小路径集。一个更具描述性的名称可能是“最小预防集”，因为它们指示了如何防止故障顶部事件的发生，并在其不发生方面取得成功。最小路径集提供了有价值的信息，说明即使没有量化，也可以防止故障顶部事件的发生。此外，ST可以量化以提供成功的概率，即不发生故障顶层事件。

此外，每个最小路径集都可以量化，以确定最有效的预防方法的优先级（通常以确保预防的成本为标准）。从失败（发生）和成功（非发生）的角度分析顶层事件的能力增加了从这些逻辑树中获得的信息的范围

### FTA决策的应用

FTA提供各种信息来辅助决策。本文概述了FTA的一些主要用途，以便读者了解FTA在决策中的广泛应用。请注意，本节包含了前面章节中已经提供的一些信息，以便读者能够关注FTA在决策过程中的作用

使用FTA来理解导致顶层事件的逻辑。故障树分析提供了导致顶层事件的基本原因和中间事件的可视化逻辑模型。通常，故障树不局限于单个系统，而是跨越系统边界。正因为如此，它们在识别影响冗余的系统交互方面显示出了巨大的好处。通过系统传播的故障和事件的组合被清楚地显示出来。最小割集可以根据所涉及事件的数量及其性质进行组织和排序。例如，如果存在只包含一个组件故障的最小割集，则表明单个组件故障可能导致系统故障。只有人为错误的故障路径表明，只有人为错误才能导致系统故障。阅读本手册后，读者应确信，从FTA获得的定性信息与所提供的定量信息同等重要。

使用FTA对导致顶级事件的贡献者进行优先排序。FTA中最重要的信息类型之一是对最重要事件的参与者进行优先级排序。如果一个FT被量化，那么导致顶级事件的故障和基本事件可以根据它们的重要性进行优先级排序。此外，中间故障和导致顶层事件的事件也可以优先排序。对于不同的应用，会产生不同的优先级和不同的重要性度量。FTA得出的一个有价值的结论是，通常只有少数几个贡献者对顶级赛事很重要。通常只有10%到20%的基本事件对最高事件概率有显著贡献。此外，贡献者常常聚集在不同的群体中，其重要性在数量级上有所不同。

从FTA得到的优先顺序可以为资源和成本的排序提供重要依据。在不影响系统失效概率的情况下，可以显著降低资源支出。对于给定的资源消耗，可以通过分配资源使其与贡献者的重要性相一致来最小化系统的故障概率。从故障树分析与从分析中得到的最高事件概率或排序割集列表一样重要。

使用FTA作为预防顶级事件的主动工具。FTA通常用于识别系统中的脆弱区域。这些脆弱区域可以在顶部事件发生之前得到纠正或改善。对系统的升级可以客观地评估其在降低顶级事件发生概率方面的益处。升级评估是FTA的一个重要用途。不同纠正措施和升级的倡导者通常会声称，他们提出的建议提供了最大的好处，而且从他们当地的角度来看，他们可能是正确的。然而，自由贸易协定是一个独特的工具，它通过系统和客观的措施衡量利益对顶级赛事的影响，提供了一个全球视角。顶部事件的概率可用于确定执行升级的关键性。顶层事件的概率可以与可接受标准进行比较，也可以用于成本效益评估。成本效益方法的进步使得不确定性和风险规避以及概率被纳入其中。此外，FTA提供的成功路径可用于确定防止顶层事件发生的具体措施。FTA的积极使用已被证明是其最有益的用途之一

使用FTA来监控系统的性能。使用FT作为监测工具是一种特殊的主动性用途，由于其特殊性，这一点已经得到确认。当监控与顶层事件相关的性能时，FTA可以解释基本事件数据的更新，以及趋势和时间相关的行为，包括老化效应。使用系统更新技术，可以用新的信息重新评估故障树，这些信息可以包括缺陷和接近故障的信息。然后可以确定维护或更换必要设备的措施，以控制故障概率和风险。FTA作为一种监测工具在核工业中很常见

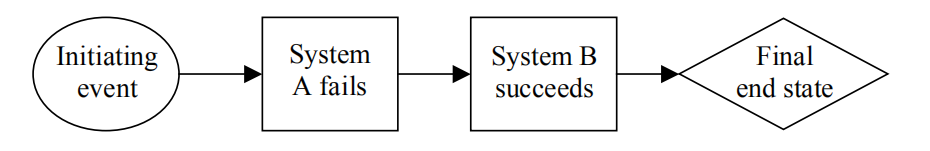
利用FTA最小化和优化资源。FTA的这种特殊用途有时被忽视，但它是最重要的用途之一。通过各种重要措施，FTA不仅确定了什么是重要的，什么是不重要的。对于那些不重要且对顶级事件的影响可以忽略不计的贡献者，可以放松资源，而对顶级事件概率的影响可以忽略不计。事实上，使用形式化的分配方法，可以重新分配资源，以产生相同的系统故障概率，同时显著降低总体资源支出。在各种应用中，FTA已被用于在不影响顶层事件发生概率的情况下，减少40%的资源负担。已经开发了软件来帮助对大型系统进行这些资源重新分配

使用故障树来辅助设计系统。在设计系统时，FTA可用于评估设计方案，并建立基于性能的设计需求。在使用故障树分析确定设计要求时，定义了性能要求，并用故障树来确定满足性能要求的设计方案。即使没有特定于系统的数据，也可以使用通用或传统数据来表示性能。这种用法FTA经常被忽视，但它的重要性足以在后续章节中进一步讨论

使用故障树分析作为诊断工具来识别和纠正顶部事件的原因。FTA作为诊断工具的使用不同于上述主动预防性应用。故障树中的顶层事件或中间事件发生时，故障树分析可作为诊断工具。当FTA能更有效地确定事件的原因或不能更有效地确定原因的优先级时。导致顶部事件的事件链在故障树中确定，提供了可能发生故障的有价值信息，以及可以纳入改进缓解措施的领域。当确定了可供选择的纠正措施时，FTA可用于客观评估其对顶层事件再次发生的影响。故障树分析还可以通过确定最有效的措施来减少故障或故障的影响，从而为应急分析提供重要帮助。在这种情况下，组件在故障树中被设置为故障状态，并识别操作以将故障的影响降至最低。这种应急分析应用程序通常用于确定如何重新配置系统，以尽量减少组件故障的影响。也可以根据部件发生的故障次数和停机次数来确定。

从上面可以看出，FTA在决策中有着广泛的用途和作用。故障树分析可用于系统从设计到系统实施和改进的整个生命周期。当系统进入寿命结束时，可以对其性能进行监控，以在发生故障之前确定趋势。当有意识地用于辅助决策时，FTA的收益通常远远超过执行分析所花费的资源

### 故障树在PRA中的作用

概率风险评估（PRA）对需要发生的事件序列进行建模，以使不期望的最终状态发生。一系列事件（事件序列）通常被称为事故序列。事故序列的一个例子是由于缓解系统无法运行而导致灾难性后果的火灾。PRA中简单事件序列的模型如下所示

注意，在上面的事件序列模型中，一个系统的成功和另一个系统的失败都会出现。哪些特定系统失败，哪些系统成功，决定了最终状态的类型及其相关后果。为了量化事故序列，需要确定事件序列中每个事件的概率，而不是最终状态。每个事件的概率取决于序列中先前的事件（例如，系统A失败的概率是给定始发事件发生时失败的概率，系统B成功的概率是给定A失败且始发事件发生时B成功的概率）。如果一个事件独立于序列中的其他事件，并且存在失效数据，则可以直接从数据中估计概率。对于序列中更复杂的事件，即没有直接适用的数据或可能与序列中的其他事件有依赖关系的事件，例如系统故障，通常会构建故障树。故障树被开发到包含系统之间依赖关系的级别，或者发展到存在基本事件的故障数据的级别，以较低的为准（更详细）。然后对故障树进行评估，以确定系统故障的概率。

每个事件序列是始发事件和除结束状态之外的后续事件的逻辑交集（与门）。可用的PRA软件使用输入到事件序列的所有故障树自动执行涉及该交叉口的操作。

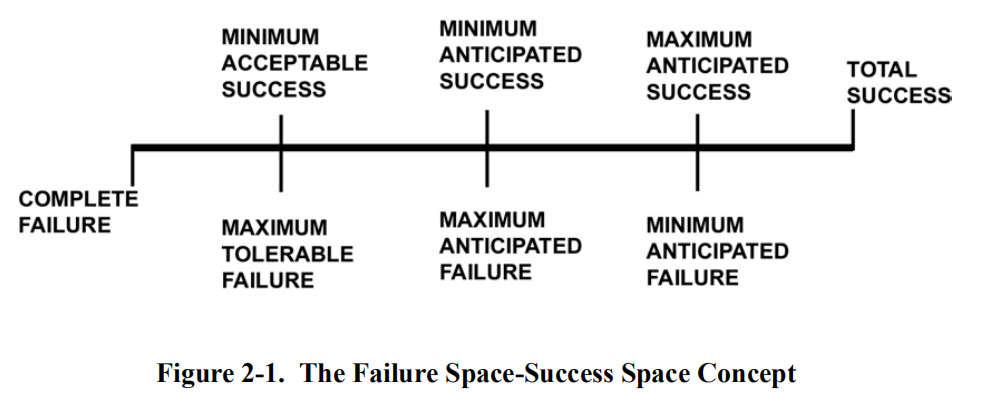
根据分辨率的不同，一个像航天飞机这样的复杂的概率风险评估可以有成千上万个事故序列，涉及数百个不同的故障树。在大型分析中，将每个序列的故障树（和门）组合成一个OR门，在单个分析运行中为整个PRA生成事故序列割集。当定义了几个不同的端状态时，每个端状态的故障树被组合起来。故障树通常是PRA的work horses，它提供了所涉及的所有系统故障的原因和概率，以及对序列进行量化的框架

### 故障树分析软件

FTA有许多软件应用程序，新的应用程序正在不断开发中。一些应用程序提供绘制和量化FT模型的能力，而其他应用程序则提供一套集成的PRA工具，包括绘制和求解FTs的能力。

## 系统逻辑建模方法

### 成功与失败的方法

一个系统的运行可以从两个角度来考虑：可以列举系统成功的各种方式，也可以列举系统失败的各种方式。此类枚举将包括完全成功的系统操作和完全的系统故障，以及中间条件，例如最小可接受成功率。图2-1描述了失败/成功空间概念

值得注意的是，成功空间中的某些可识别点与失败空间中的某些类似点重合。因此，例如，成功空间中的“最大预期成功（maximum anticipated success）”可以被认为与失败空间中的“最小预期失败（minimum anticipated failure）”一致。

虽然第一种倾向可能是在考虑系统运行时选择乐观的观点（成功）而不是悲观的观点（失败），但这并不一定是最有利的观点。

从分析的角度来看，从失效空间的角度来看，有几个压倒一切的优势。首先，在什么是失败上达成一致意见通常比就什么是成功达成一致意见要容易得多。一架飞机可能需要飞得又高又快，不用加油就能飞得很远，而且能运载大量的货物。当这架飞机的最终版本下线时，在进行设计权衡的过程中，这些特性中的一些可能已经被破坏了。这辆车是否“成功”可能是一个很有争议的问题。另一方面，如果飞机坠毁，几乎没有理由认为这起事件构成了系统故障。

“成功”往往与一个系统的效率、产出量、有用程度以及生产和营销特征相联系。这些特征可以用连续变量来描述，这些变量不容易用简单的离散事件来建模，例如“阀门不打开”，这些事件表征了故障空间（部分故障，即阀门部分打开，由于其连续的可能性，也很难建模）。

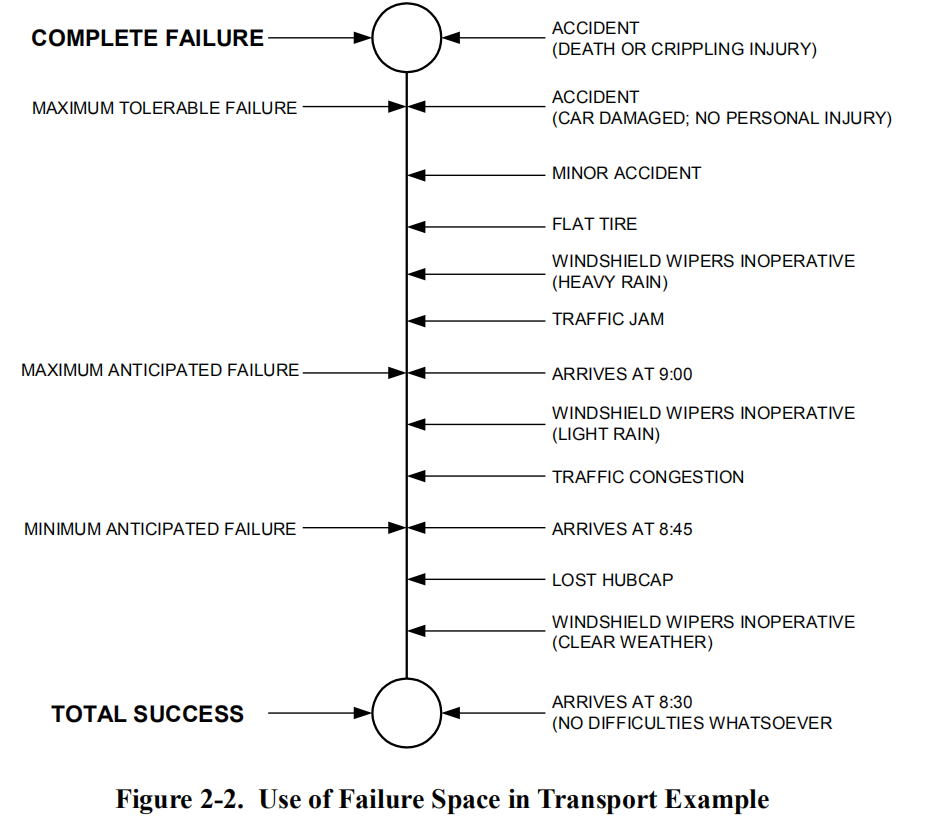
因此，事件“失败”，特别是“完全失败”，通常很容易定义，而事件“成功”则可能更难界定。这使得在分析中使用失败空间比使用成功空间更有价值。

另一个有利于使用失败空间的观点是，虽然理论上一个系统可能失败的方式和一个系统能够成功的方式的数量都是无限的，但是从实际的角度来看，成功的方式通常比失败的方式多。因此，纯粹从实用的角度来看，失败空间中的总体规模小于成功空间中的总体规模。因此，在分析中，基于失效空间进行计算通常更有效。

支持使用失效空间的最后一点是失效模型量化所涉及的数学性质。当组合失效概率小于0.1时，最准确的失效概率小于1。由于成功概率通常接近1.0，因此不能使用这些近似值，因此在组合成功概率时必须使用复杂的计算。因此，成功模型的解决方案与失败模型的解决方案大不相同。

在过去的许多场合中，在分析系统运行时，故障空间的有利使用已经被证明是可行的。绘制复杂系统的逻辑图是一项耗费大量时间的工作。当考虑到故障时，可能只需要构建一个或两个系统模型，例如故障树，覆盖所有重要的故障模式。当考虑成功时，可能有必要构建几百个系统模型，涵盖各种成功定义。Minuteman导弹分析是故障空间事件简化的一个很好的例子。仅绘制了三个与三个意外事件相对应的故障树：意外程序发射、意外发动机点火和故障发射。人们发现，仅对这三起事件进行仔细分析，就可以全面了解复杂的民兵系统。

考虑图2-2中的“任务”指的是用汽车将X个人从家里送到办公室。预期到达时间是8:30，但是如果X在9:00之前到达办公室，任务将被视为稍微成功。在8:30到达被标记为“最小预期故障”。在“最小预期故障”下面是一些可能的事件，这些事件构成了小麻烦，但并不妨碍X在期望的时间到达。

9:00到达时被标记为“最大预期故障”。在这一点和“最小预期故障”之间，有许多事件导致X的到达时间延迟半小时或更少。也许合理的做法是，让“最大可容忍故障”这一点与某些事故相吻合，这些事故会对汽车造成一些损坏，并造成相当大的延误，但不会造成人身伤害。在这一点之上，越来越严重的事件最终导致完全失败或死亡

请注意，“挡风玻璃雨刮器不工作”等事件将根据当时环境的性质沿线路定位。

如图2-2所示的图表也可用于确定商业客机生产中的事件。“最小预期故障”一点对应于所有规范的实现情况，以下各点表示某些规范已经超过了要求。“最大预期故障”这一点对应于一些权衡点，在该点上，所有规范都没有得到满足，但这些差异并不严重，不足以在实质上降低飞机的可销售性。

“最大容许故障”点对应于制造飞机的公司的生存点。超过这一点，只有无法忍受的灾难才会发生。一般而言，但并非所有情况下，故障树分析致力于识别和评估此类灾难性事件和完全故障

### 演绎方法和FTA

演绎构成从一般到具体的推理。在演绎系统分析中，假设系统本身以某种方式失效，并试图找出系统或子系统（组件）行为的模式是导致这种故障的原因。一般来说，这种方法可以称为“夏洛克-福尔摩斯”的方法。福尔摩斯面对所提供的证据，其任务是重建导致犯罪的事件。事实上，所有成功的侦探和其他类型的调查员都是演绎分析方面的专家。

在现实生活中，典型的演绎分析是事故调查：是什么一连串的事件导致了像泰坦尼克号这样的“永不沉没”的船在处女航中沉没？什么样的故障过程，仪器和/或人为因素，造成了垂直安定面的损失和一架商用客机坠入居民区？

这本书的主要主题，故障树分析，是一个演绎系统分析的例子。在这项技术中，假定一些特定的系统状态，通常是一种故障状态，并以系统和逻辑的方式建立起导致这种意外事件的更基本的故障链。FTA的一般原则，以及FTs的应用和评估的细节，将在后面的章节中给出。

虽然演绎法被用来确定一个给定的系统状态（通常是一个失败的状态）是如何发生的，但是归纳法可以用来确定哪些系统状态（通常是失败的状态）是可能的。归纳法在风险和可靠性分析中发挥着重要作用，特别是在事故场景的制定和确保完整性方面。

归纳法将在下一节和第9章中描述，并与中的FTA进行比较 第2.4条。

### 归纳法

归纳法涉及从个别案例到一般结论的推理。如果考虑到某一系统，假设某一特定故障或启动条件，并试图确定该故障或条件对系统运行的影响，则正在进行感应系统分析。因此，可以对某些特定控制面的损失如何影响飞机飞行或预算中某些项目的取消如何影响学区的整体运作进行调查。

在日常语言中，归纳技术提供了对一般问题的答案，“如果…？“更正式地说，这个过程包括假设组件的特定状态、通常是失效状态，然后分析对系统的影响。更一般地，假设给定的启动事件，例如管道破裂，并分析事件的后果。这种更一般的方法用于事件树和事件序列图，在第9章中讨论。

归纳方法也称为自底向上的方法，即在故障发起者和基本事件发起者处，然后向上确定给定启动器的系统影响。归纳法由此从可能的基本原因开始，然后分析结果的影响。分析了一系列可能的原因，以期对其产生影响。

归纳方法的主要示例如下所述。

#### “部件计数”方法（The “Parts Count” Approach）

对于一个系统，最简单、最保守（即悲观）的假设可能是，任何单个部件的故障都会导致整个系统的故障。在这种假设下，获得系统失效概率的上界尤其简单。列出了所有部件及其估计的故障概率。

然后将单个部件的概率相加，这个和提供了系统失效概率的上界。故障概率可以是故障率、不可靠性或不可用性，具体取决于特定的应用程序（这些更具体的术语将在后面讨论）。

对于一个特定的系统，部件计数技术可以提供一个非常悲观的系统失效概率估计，悲观程度通常无法量化。“零件计数”技术是保守的，因为如果存在关键部件，它们通常会出现冗余，因此没有一个故障实际上对系统是灾难性的。此外，一个部件通常会以几种不同的方式偏离其正常运行模式，这些故障模式一般不会对系统运行产生同样的有害影响。如果不知道系统运行的相关故障模式，则有必要将所有可能的故障模式的故障概率相加。零件计数法没有进一步讨论，而是作为最简单的归纳法引入，其中假设每个部件故障都会导致系统故障。

#### 失效模式与影响分析（FMEA）

表2-1显示了由冗余放大器系统构成的FMEA。在构建表格时，我们认识到放大器可能以多种方式发生故障，而首要任务是识别这些不同的故障模式。两个主要模式是“开路”和“短路”，但假设分析还检测到28种其他模式（例如，弱信号、间歇性接地等）。任何放大器的短路是一种更为关键的故障模式，因为它总是会导致系统的故障。FMEA表包含以下信息：

（1）部件名称

（2）故障概率（故障率或不可用性是使用的一些特定特性）

（3）部件故障模式

（4）每种模式导致的总故障百分比

（5）对整个系统的影响，分为不同的类别（最简单的两个类别是“关键”和“非关键”）。

根据这类放大器的以往经验，估计90%的放大器故障可归因于“打开”模式，5%归因于“短”模式，其余5%归因于“其他”模式。众所周知，每当任何一个放大器发生故障短路时，系统都会发生故障，因此X被置于这些模式的“关键”列中；“关键”意味着单一故障导致系统故障。另一方面，当任一放大器失效时，由于并联结构，单一故障对系统没有影响。其他28种故障模式的严重程度如何？这个例子是保守的，因为所有其他故障模式都被认为是关键的，即任何一种模式的发生都会导致系统故障。临界列中显示的数字是通过将第4列中的适当百分比乘以第2列的1×10-3得到的

根据该表，仅考虑那些关键的失效模式，可以更实际地计算单一原因导致的系统失效概率。将临界列5相加，得到系统失效概率=5×10-5+5×10-5+5×10-5+5×10-5=2×10-4。与未分离临界失效模式的零件计数法得出的2×10-3相比，这是一个不太保守的结果。两个系统结果之间的差异可能很大，即一个数量级或更多，如在本例中，如果临界失效模式占总失效模式的一小部分（例如，10%或更少）。

FMEA（及其变体）可以合理确定地识别具有“非关键”影响的部件故障，但实际考虑的可能部件故障模式数量有限。保守主义规定，未指明的故障模式和可疑影响被视为“关键”（如前一个例子所示）。分析的目标是识别单一故障模式并量化这些模式；分析不需要比这些目标所需的更详细

#### 关键性和失效模式分析(FMECA)

失效模式影响和危害性分析（FMECA）本质上类似于FMEA，其中更详细地分析了故障的严重性，并描述了限制此类故障可能性的保证和控制措施。尽管FMECA不是一种检测危险的最佳方法，它经常被用于系统安全分析过程中。这种方法的四个基本方面是

（1）错误识别

（2）故障的潜在影响

（3）现有或预计补偿和/或控制

（4）结果总结。

这四个方面通常作为FMECA布局中的列标题出现。第1列确定了可能的危险情况。第2列解释了为什么这种情况是一个问题。第3列描述了如何补偿或控制这种情况。最后，第4栏说明局势是否得到控制或是否应采取进一步措施。

在这一点上，读者应该得到警告，在所有这些归纳技术中或多或少都存在着一个最危险的陷阱：将形式误认为物质的可能性。

如果项目仅仅是填写表格而不是进行适当的分析，那么这个练习将是完全徒劳的。因此，最好不要把分析局限于任何事先准备好的形式主义。另一点：如果系统完全是复杂的，那么想象一个分析员就可以对所有系统故障及其对系统的影响进行正确而全面的调查是愚蠢的。这些技术需要一个良好协调的团队方法。此外，FMEA和FMECA分析单一部件故障及其系统影响，而不考虑部件故障的组合

#### 初步危害分析(PHA)

到目前为止所描述的技术在很大程度上是面向系统的，即对系统运行的影响是故障。本节的主题，初步危险分析（PHA）是一种评估系统对工厂人员和其他人构成的潜在危险的方法。

PHA的目标是识别系统内固有的潜在危险条件，并确定可能发生的潜在事故的重要性或重要性。PHA研究应尽早在产品开发阶段进行。这将允许早期制定控制这些危险条件的设计和程序安全要求，从而消除以后昂贵的设计变更。

PHA的第一步是识别系统中潜在的危险元素或组件。工程经验、工程判断的运用以及不时制定的大量检查表的使用促进了这一过程。PHA的第二步是识别可能将特定危险条件转化为潜在事故的事件。然后对这些潜在事故的严重性进行评估，以确定是否应采取预防措施。

为了促进PHA过程，开发了各种列格式。也许最简单的是：

第（1）列：部件/子系统和危险模式

第（2）列可能的影响

第（3）栏补偿和控制

第（4）栏调查结果和备注

#### 故障危害分析(FHA)

另一种方法，故障危险分析（FHA）是一种特殊用途的工具，用于涉及多个组织的项目，其中一个组织被认为是集成器。这种技术对于检测跨组织接口的故障特别有价值。尽管FHA本身目前一般不使用，但某些扩展的FMEA和FMECA中使用了FHA的概念和方法。FHA方法考虑了以下基本原因和影响，这些因素可以按列排列，这些因素是归纳法的特点：

第（1）列部件标识

第（2）列失效概率

列（3) 故障模式(识别所有可能的模式）

第（4）列 按模式列出的故障百分比

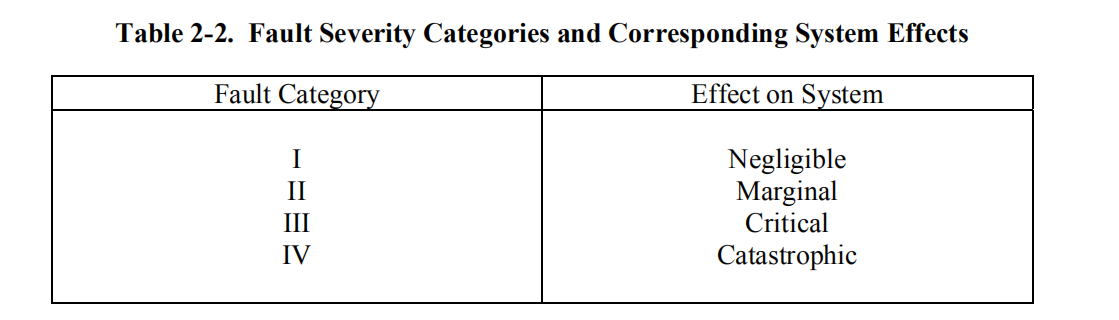
第（5）列 故障影响（追溯到相关接口）

第（6）列 可命令或启动相关故障的上游组件的标识

第（7）列 可能导致二次故障的因素（包括阈值水平）。此列应包含组件敏感的操作或环境变量列表第（8）栏备注

FHA的不同之处在于考虑了第6列和第7列中给出的额外信息。第6列指出了可能的命令或接口故障。第7列列出了设计范围之外的次级故障。在后面的章节中会很明显，第6列和第7列对于故障树分析员具有特殊意义。

#### 双重失效矩阵（DFM）

以前的技术关注单一故障的影响。双重故障矩阵（DFM）是一种同时考虑双重故障影响的归纳技术，它适用于具有少量冗余部件的可行系统。DFM方法有助于讨论，因为它将归纳方法从单个故障原因扩展到多个故障原因。这是对FMEA和FMECA方法的重大改进。为了更有效地应用DFM方法，首先根据系统影响的严重程度对包括多个故障在内的故障进行分类。源于MIL STD 882且仍在使用的基本分类如表2-2所示。

更完整的系统效应定义是：

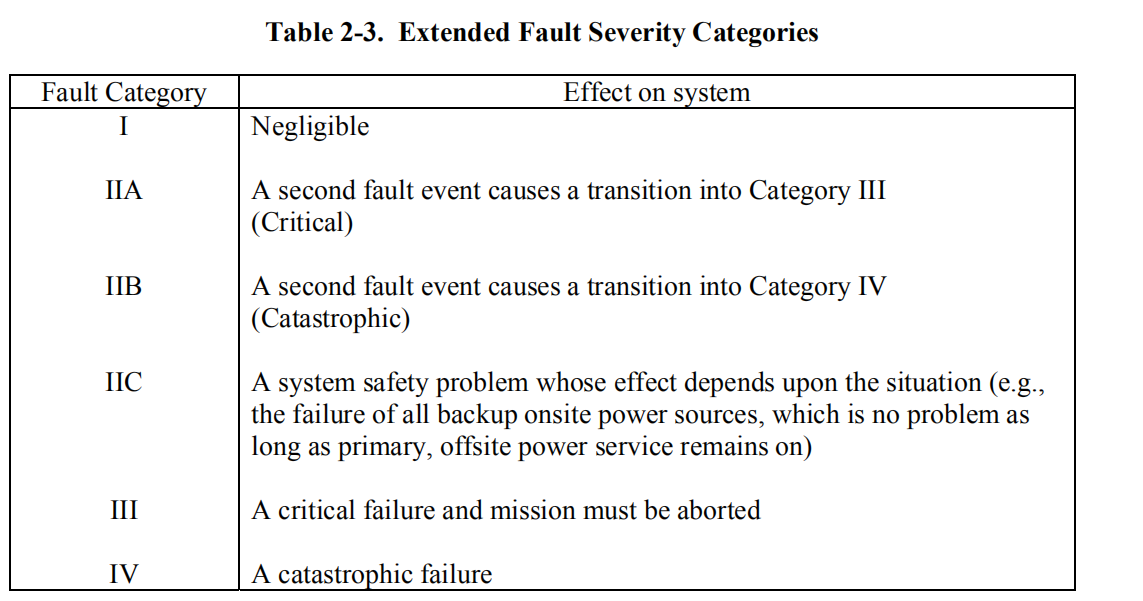
（I）可忽略的-对系统没有影响的功能损失。

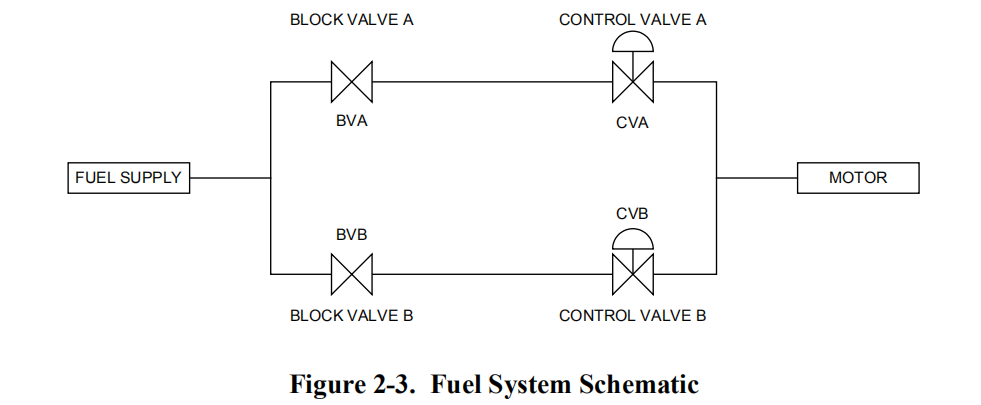
（二） 边际-此故障将在一定程度上降低系统性能，但不会导致系统不可用；例如，两个冗余泵中的一个失去，其中任何一个都可以执行所需的功能。

（三） 严重-该故障将完全降低系统性能；例如，部件的丢失会导致安全系统不可用。

（四） 灾难性-该故障将产生严重后果，可能导致人员伤亡；例如，灾难性压力容器故障。

分类将取决于先前假设存在的条件，并且分类可以随着假设条件的变化而变化。例如，如果假设一个泵发生故障，那么第二个冗余泵的故障就是严重故障。

以上故障分类可以通过多种方式进行细化。如表2-3所示为6个故障类别。这些故障类别最初在NERVA项目中使用，但在最近几年中使用，只是稍作修改

为了说明DFM的应用，请考虑图2-3所示的简单子系统。在此图中，切断阀只能在全开或全关状态下运行，而控制阀是比例阀，可以部分打开或部分关闭。

将该系统的两种故障状态定义和分类如下：

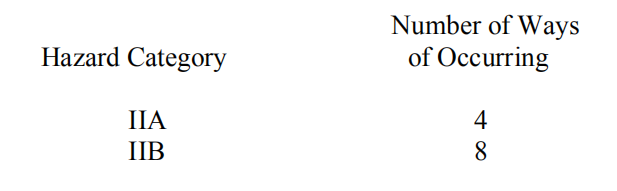
无法打开-Ⅳ

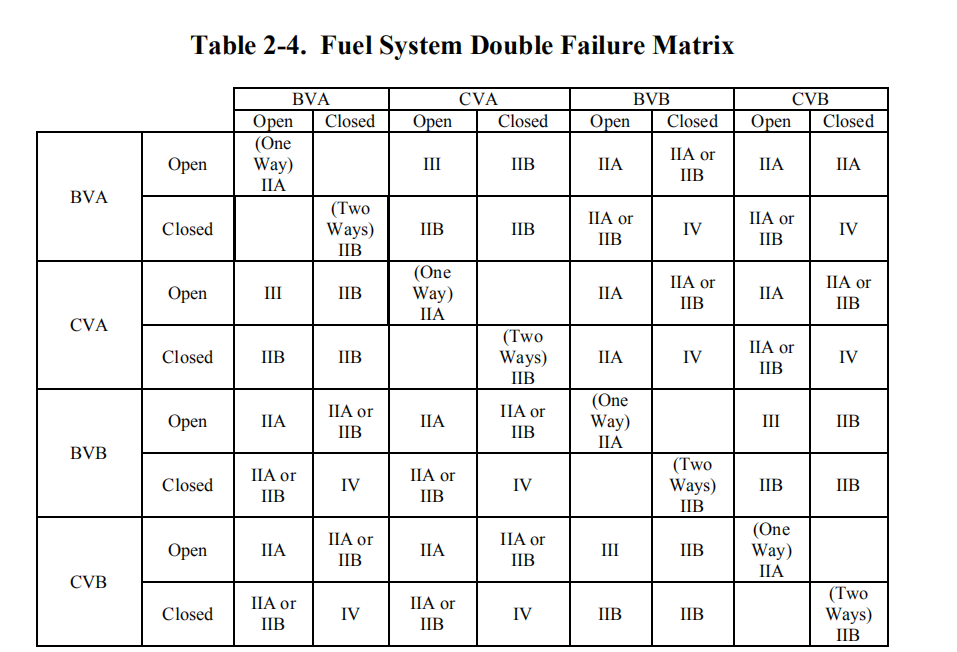
阀门无法关闭-III

现在考虑所有可能的部件故障及其故障类别。例如，

切断阀A（BVA）无法打开被认定为IIA类，因为如果控制阀A（CVA）也无法打开，则会产生III类。BVA关闭失败被标识为IIB类，因为如果BVB或CVB也未关闭，则会产生IV类结果。这种类型的分析可以方便地在表2-4所示的双重失效矩阵中系统化。

为了便于说明，已填写了整个矩阵；对于一阶分析，只关注主要的对角线项，即单一失效状态。注意，如果BVA是失效开放的，第二个故障只能通过一种方式级联到第三类；也就是说，CVA也必须失效打开。相反，如果BVA关闭失败，则BVB或CVB中的任何一个也关闭失败，则会产生第四类，这就是为什么表2-4给出了“两种方式”。类似的考虑也适用于CVA、BVB和CVB的单一故障，这些重要的附加信息已显示在矩阵的主对角线单元格中。

现在只关注单个故障，进行危险类别统计，如下表所示：

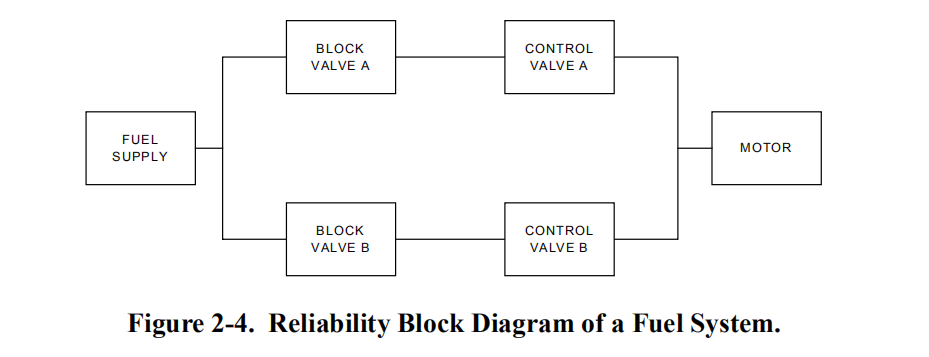


如何使用这些信息？其中一个应用是对这些危险类别如何控制或投保的描述和随后的审查。另一个应用是图2-2所示阀门配置与替代设计之间的比较。DFM是一种有用的冗余归纳方法，但它在处理小组合，特别是双故障方面的能力有限，只能评估有限数量的组合。软件允许更多的组合和超过两个的多个故障被评估。但是，仍然必须对所考虑的组合施加约束

#### 可靠性框图（RBD）

可靠性框图是一种归纳模型，其中系统被划分为表示不同元素（如组件或子系统）的块。然后，根据系统成功路径组合这些元素块。RBD通常用于表示系统中的活性元素，其方式允许对所有成功途径进行彻底搜索和识别。元素之间的依赖关系可以显式地解决。

最初开发的顶层rbd可以连续分解，直到获得所需的详细程度。或者，在详细的RBD中代表系统序列的系列组件可以直接或通过使用FTs逻辑组合成一个超级组件，然后与其他超级组件链接，形成系统的概要模型。这种表示有时会导致更透明的分析

图2-3所示燃油系统的RBD示例如图2-4所示。代表每个系统元件（燃料供应、切断阀、控制阀和电机）的独立块在结构上组合起来，以表示通过系统的两个潜在流动路径。该模型通过列举系统中不同的成功路径来求解，然后使用布尔代数规则将这些块延续成系统成功的整体表示。

软件可将RBD转换为FTA，反之亦然。虽然这些转换会产生逻辑上等价的模型，但转换中的逻辑表示可能不像原始模型中那样清晰

### FTA与归纳法的比较

FTA与归纳法的根本区别在于分析的方向。故障树分析从不希望发生的事件开始，追溯到必要和充分的原因。

故障树以 初始基本事件和确定为主要原因的故障结束。归纳法从 一个始发原因开始，并追溯由此产生的后果。对于不同的选择的始发原因，重复这一前进步骤。最终结果可能因始发原因而异。因此，如果确定了给定的意外事件，并且目标是确定其基本原因，则FTA是进行的适当分析。

归纳分析是一种恰当的分析方法，如果确定了一组给定的始发原因，并且目标是确定由此产生的后果。审慎监管局同时使用归纳法和演绎法。始发事件通常使用一种称为主逻辑图（MLD）的演绎方法来识别。参考文献1描述了这一概念。对于可能发展到多个故障状态的始发事件，必须单独跟踪（即空间航天飞机LOV），PRA使用一个归纳工具，如事件树，不仅确定结果的后果，而且列举所有可能的事故情况。系统分析通常评估已定义的不良系统事件（如系统故障）的原因。因此，FTA是系统分析的合适的分析工具

一般来说，为了得到一套完整的事故序列，必须采用演绎和归纳两种方法。演绎法的优点是将分析重点放在不希望发生的事件上，而归纳法有助于确保分析足够广泛，能够涵盖所有可能的情况。

FTA和归纳分析当然有重叠之处。如前所述，在PRA中使用故障树来分析事件序列中的特定系统故障。可以为不同的意外事件构建多个故障树，并且通常是为评估一系列后果而构建的。在这些应用中，FTA与归纳分析的基本区别在于分析的方向，FTA是逆向分析，归纳分析是正向分析

## 故障树分析详解

### 故障树分析的步骤

成功的FTA需要进行以下步骤：

1。确定FTA的目标（objective）。

2定义FT的顶级事件(top-event)。

3。定义FTA的范围(scope)。

4定义FTA的解决方案(resolution)。

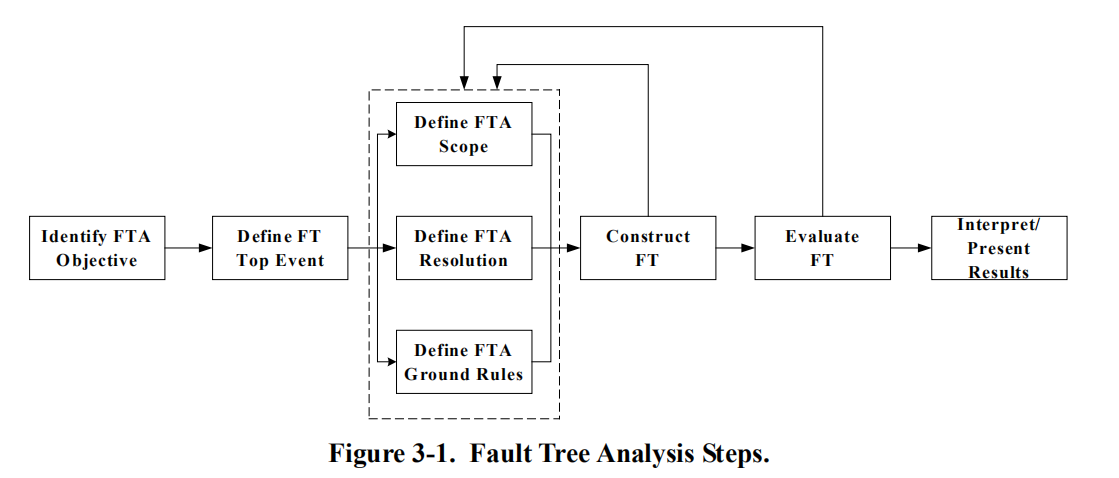
5定义FTA的基本规则(ground-rules)。

6建造FT。

7评估FT。

8解释并呈现结果

前五个步骤涉及FTA问题的制定。剩下的步骤包括FT的实际构造、FT的评估和FT结果的解释。

虽然大多数步骤是按顺序执行的，但是步骤3-5可以同时进行。在步骤6和7中修改步骤4和5的情况并不少见。这八个步骤的相互关系如图3-1所示。反馈如图所示

成功的FTA首先要明确FTA的**目标**。这似乎是显而易见的；然而，有很多情况下进行了故障树分析，但分析并不能满足委托它的决策者或管理者的目标。为了成功，目标应该用要分析的系统的失败来表述。例如，如果总体目标是评估任务的不同设计，则需要确定任务失败的特征以及将被分析以评估设计的特定故障。

一旦以这种方式定义了目标，那么FT的**顶部事件**也被定义（步骤2）。

故障树的顶部事件是故障原因将被解决并确定故障概率的事件。顶层事件定义将要分析的系统的故障模式。一个例子就是发动机的良性关闭。有时，目标可能需要定义和分析不止一个失败。在这种情况下，将定义单独的top事件。特别是，对于一个给定的任务，可能会有多个目标，并且根据为分析选择的特定目标，生成的故障树可能会非常不同。例如，对于载人发射器，目标可能是车辆损失（LOV）、乘员损失（LOC）或任务损失（LOM）。对于无人驾驶火星样本返回任务，目标可能是样本控制保证（CA）或任务成功（MS）。在每种情况下，即使对于同一个任务，生成的故障树也可能不同。如果任务有不同的阶段，那么可能需要为每个阶段定义一个单独的顶层事件。分阶段任务分析将在下一节讨论。

在步骤3中，定义了分析的**范围**。故障树分析的范围表明哪些故障和原因将被包括，哪些不包括在内。FTA的范围还包括特定的设计版本和与将要分析的系统相关的历史时间段。最后，范围包括分析的边界条件。边界条件包括元件的初始状态和系统的假定输入。FT表示给定配置和边界在给定时间的系统快照。例如，假设正在分析飞行控制系统的故障。在定义范围时，应确定待分析系统的版本、定义的运行模式、要考虑的部件故障，并确定将为其故障建模或假定未发生故障的系统接口（例如，支持系统、驱动信号）

在流程的第4步中，定义了FTA的**解决方案**。解决方案是顶层事件失效原因的详细程度。如果顶部事件是系统的功能故障，例如操作失败或意外关闭，则顶部事件通常解决在系统中的主要部件上。主要部件的例子有阀门、泵和控制模块。如果顶部事件是现象学故障，例如发动机的灾难性爆炸，那么解决方案就是模拟爆炸原因的详细程度。定量模型的开发是基于对最高事件概率的最佳估计的需要，考虑到可用的数据和其他信息。故障树被发展到一个细节水平，在那里最好的失效概率数据是可用的。当对子部件或支持系统做出决策时，或当分析中无法证明某个事件独立于其他事件时（例如，与其他系统具有共同的驱动信号或功率的系统），需要进一步解决系统问题。故障树的详细程度可以，而且通常是低于可用数据估计基本事件概率的水平，或者风险识别不再重要的水平。这通常是不必要的。在后面的章节中，FTs将演示不同类型系统和顶级事件的分辨率级别。

在步骤5中，定义了FTA的所有**基本规则**。这些基本规则包括在FT中命名事件和门的程序和命名法。使用的命名方案对于创建可理解的FT非常重要。给给出了将被说明的FT的命名方案的例子。此外，还可以给出在FT中对特定故障进行建模的基本规则。这些建模基本规则有助于在不同的FTs之间提供一致性，尤其是在不同的个人开发这些FTs时。建模基本规则可以包括对特定组件故障、人为错误和共因故障进行建模的方式。在后面的章节中，我们将描述一些可以使用的建模基本规则。将要介绍的FTs还将说明一些已应用的建模基本规则。

步骤6涉及FT的**实际构造**。随后的章节将详细描述从系统原理图和描述中构建故障树所涉及的思想和逻辑。文中还描述了故障树中用来表示事件之间关系的符号。

第7步包括FT的**评估**。评估包括定性和定量评估。定性评估提供了顶部事件的最小割集信息。还可以确定成功集，以确保预防顶级事件。基本事件的性质和组合集合中基本事件的数量提供了有关顶层事件发生的重要信息。割集通常按割集顺序（割集中的事件数）排序，以提供有关可能导致顶级事件的基本事件组合的信息。定量评价不仅产生了顶事件的概率，而且产生了影响顶事件概率的主导割集，以及每个基本事件对顶事件概率的定量重要性。

这种情况下的割集按概率排序，低概率割集从分析中截断。（当需要计算效率时，有时不需要求解割集来计算顶事件概率。）针对不同的应用确定了不同的数量重要性。敏感性研究和不确定性评估提供了进一步的关键信息。

最后，步骤8涉及**结果的解释和表示**。强调的是解释，而不是简单的陈述。许多自由贸易协定未能产生重大影响，因为其结果只是简单地记录在一份报告中。必须对结果进行解释，以提供切实的影响，特别是对目标的潜在影响。如果决策者或管理者只得到一组数值和一些行话，那么分析可能没有什么影响，很可能会阻止管理者尝试未来的FTA！

### 构建故障树的基本范式

构建故障树的基本范式是“小思考”，或者更准确地说是“短视思考”。对于所分析的每个事件，确定导致该事件的必要和充分的即时事件（即最密切相关的事件）。关键短语是“必要和充分的即时事件”。分析没有跳到事件的基本原因。取而代之的是采取一个小步骤，并确定导致事件的直接事件。这一小步的倒退确保了所有的关系和主要原因都会被发现。这个最小的步骤也使分析员能够洞察发生故障树顶部事件所必需和充分的关系。这种倒退以确定构成分析解决方案的基本原因结束。

本手册的后续章节将描述构建故障树所涉及的反向、即时步骤。然而，一个简单的例子可以说明这个范例。考虑一下水龙头没有水流出的情况。造成这种情况的直接原因不是自来水公司关了水或家外的水管破裂（尽管这些最终可能是主要原因），直接原因要么是水龙头出故障（即堵塞），要么就是没有向水龙头供水。如果检查水龙头并发现其可操作，则检查水龙头的水管线是否堵塞或破裂。如果不是，则检查管线中的下一个障碍物，如上游阀门。如果没有水从阀门流出，则直接原因是阀门出现故障或没有向阀门供水。在确定主要原因之前，将执行反向立即步骤。这个思维过程说明了故障树分析的范例。

### 分析的边界

与任何建模技术一样，FTA的边界必须被定义。这应该在构建故障树之前首先完成，并且必须记录在案。在构建故障树时，边界可能会发生变化，这些变化必须记录在案。如果将系统故障分析为意外事件，那么定义分析边界涉及定义将要分析的系统边界。与系统的接口（如电源或供水）通常包含在分析中，因此在分析边界内。如果它们被排除在分析之外，则需要定义它们的状态，以便定义要分析的组件的输入。

更一般地说，定义分析的边界涉及定义分析中的内容和分析之外的内容。在分析中的是那些参与者和事件，它们与最不希望发生的事件之间的关系将被分析。分析之外的是那些没有被分析的贡献者。在影响被分析的贡献者的意义上，将有贡献者处于边界。这些接口的状态需要根据对被分析的贡献者的输入的假设来定义。

通常在故障树上显示接口及其假定状态是一种很好的做法。这有助于记录分析的边界。同样重要的是，它是允许在以后的阶段开发接口，还是允许在其他地方开发的接口模型连接到故障树中

### 顶级事件的定义

故障树的顶部事件指导所有其余的分析。如果顶层事件的定义不正确（这种情况发生的次数惊人），那么FTA将是错误的，这可能导致错误的决策。由于分析员认为故障树顶部事件的定义是正确的，但没有与决策者核实或将定义与程序目标相关联，因此错误地定义了故障树的大量案例。这就是为什么定义和理解分析的目标和要解决的问题是极其重要的。定义几个潜在的顶级事件，然后根据与决策者和其他相关人员的协商来决定适当的一个或多个事件通常是富有成效的。在定义顶级事件时，根据定义事件发生的特定标准来定义事件是很重要的。通常要对系统故障执行此操作，首先定义系统成功标准。然后将系统的失败定义为未能满足给定的成功标准。例如，如果给定冷却系统的成功标准是三台泵中的两台启动并运行12小时，则系统故障是两台泵无法启动或无法继续运行12小时。这是顶级事件定义。系统的初始状态也需要定义为顶层事件的一部分或分析范围的一部分。这些观点可以概括如下：

1要定义顶级事件，请定义事件发生的条件。对于系统故障，首先定义系统成功标准。

2确保顶层事件与待解决问题和分析目标一致。

3如果不确定顶层事件，定义涵盖顶层事件的其他定义，并评估每个定义的适用性。

### Faults vs. Failures

这里对相当具体的单词“failure”和更一般的单词“fault”进行了区分。如果继电器在其端子上施加电压时正确闭合，这就是继电器“success”。但是，如果在这种情况下，继电器不能闭合，这就是继电器“failure”。另一种可能性是，由于某些上游部件的不正常工作，继电器在错误的时间关闭。这显然不是继电器failure；但是，继电器操作不及时很可能导致整个电路进入不满意的状态。这种情况在这里称为“fault”，因此，一般来说，所有failure都是fault，但并非所有fault都是failure。failure是基本的异常事件，而fault是“高阶”或更一般的事件。

接下来考虑一座桥，它应该偶尔开放，以允许海上交通通行。突然，在没有任何警告的情况下，桥的一片叶子因为操作员的命令而翻转了几英尺。这不是网桥故障，因为它应该在命令下打开，但它确实打开了。但是，该事件是一个错误，因为网桥机制响应了网桥助理发出的不及时的命令。因此，乘务员是这个“系统”的一部分，正是他的不合时宜的行为导致了桥梁故障。

对故障的正确定义不仅需要说明不需要的组件状态是什么，还需要说明它何时发生。这些“什么”和“何时”规范应该是输入到故障树中的事件描述的一部分。

根据系统的性质，故障可能是可修复的还是不可修复的。在不修理的情况下，发生的故障将继续存在。在可修系统中，必须区分故障的发生和故障的持续存在。这种区别只在故障树量化中有重要意义（在后面的章节中讨论）。从构造故障树的角度来看，只关注发生的现象。

### 失效机理、失效模式和失效效果（Failure Mechanism, Failure Mode, and Failure Effect）

系统、子系统和组件的定义是相对的，并且依赖于分析的上下文。“系统”是所考虑的总体结构，它又由称为“子系统”的从属结构组成，而这些子结构又由称为“组件”的基本构件组成

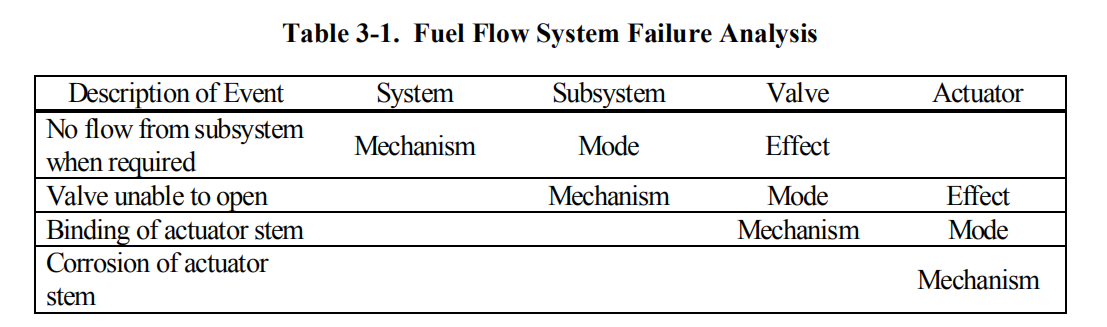
例如，在航天飞机固体火箭助推器（SRB）的故障树中，推力矢量

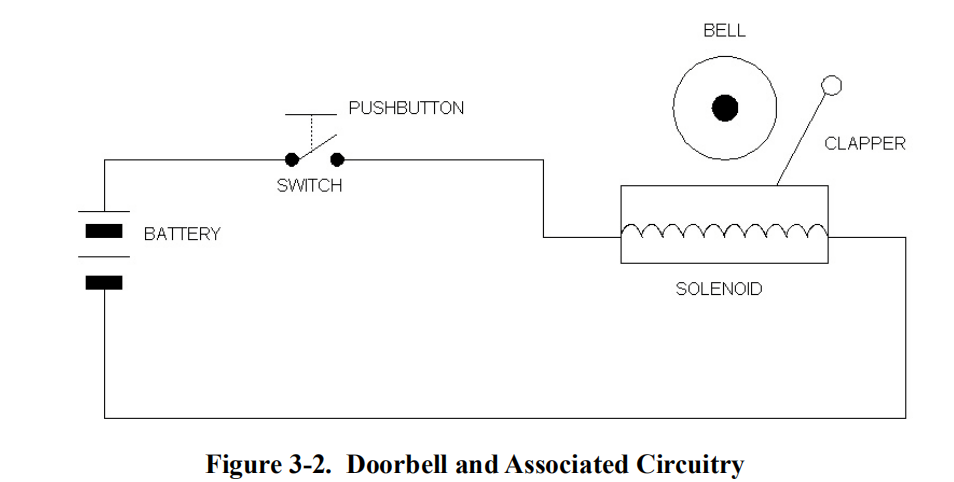
控制（TVC）可称为一个系统。例如，子系统就是辅助系统

TVC的动力装置（APU）。一个部件就是APU的燃油泵。在特定的分析中，系统、子系统和组件的定义通常是为了方便，以便为问题提供层次结构和边界。这些定义也用于为故障树开发的命名方案中。FTA的一个关键方面是从功能的角度来看待系统的各个要素，人工系统的边界在很大程度上被忽略了。

在构建故障树时，失效效应、失效模式和失效机制的基本概念对于确定事件之间的适当相互关系非常重要。当解决故障影响时，关注的是为什么某个特定的故障会引起关注，即它对系统的影响（如果有的话）是什么？当详细说明故障模式时，部件故障的哪些方面值得关注？当列出故障机制时，如何发生特定的故障模式？因此，失效机制是失效模式发生的手段，而失效模式又是更基本原因的影响。或者，故障机制会产生故障模式，而这些模式又会对系统运行产生一定的影响。

为了说明这些概念，请考虑一个控制燃料流向发动机的系统。在表3-1。感兴趣的子系统包括一个阀门和一个阀门执行器。可以从系统、子系统或组件的角度对可能发生的各种事件进行分类。

下表左栏给出了一些事件。例如，“阀门无法打开”是子系统故障的一种机制，是阀门故障的一种模式，也是执行机构故障的一种影响

为了更清楚地区分机械模式效果，从系统、子系统和组件设计者的角度考虑门铃及其相关电路的简单系统。系统示意图如图3-2所示

从系统设计者的角度来看，系统的故障模式是：

（1）按按钮时门铃不响。

（2） 不按按钮时门铃不小心响了。

（3） 松开按钮时门铃不响。

如果系统设计师现在坐下来，列出引起相关故障模式的故障机制清单，则会生成一份与实际采购开关、电铃螺线管单元、电池和电线的子系统设计师的故障模式相对应的清单。

这些是：

（1）开关-

（a）无法接通触点（包括意外断开）

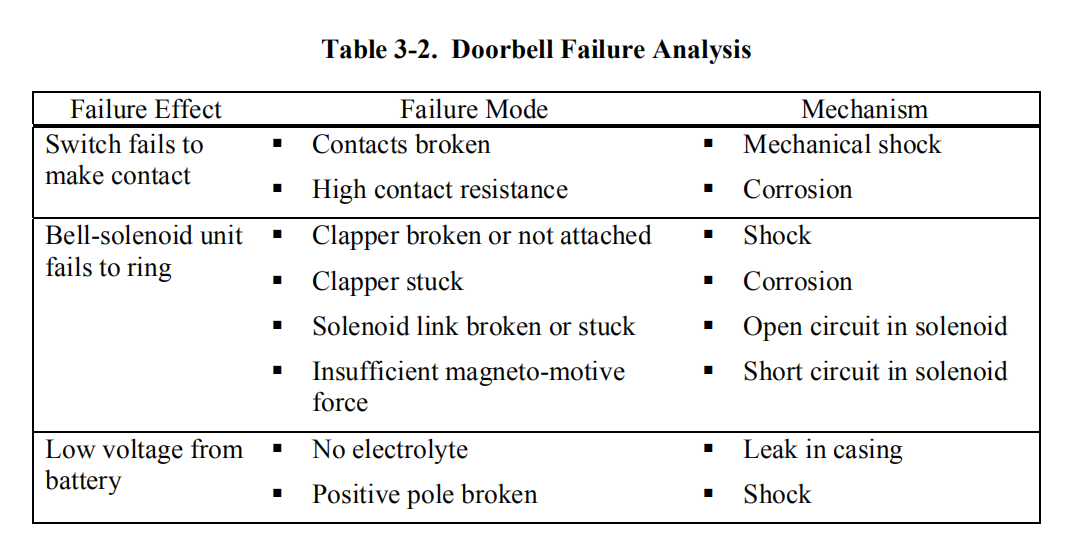
（b）无法断开触点

（c）意外闭合

（2）电铃螺线管单元在通电时无法响铃（包括在通电的情况下无法继续响铃）

（3）蓄电池低电压状况

（4）导线断路或短路

再次强调最后一个列表构成了系统设计者的失效机制和子系统设计者的失效模式。从组件设计者的角度来看，这也是一个失败影响的列表。试着想象一下组件设计者会列出什么样的列表。见表3-2。

系统失效模式构成了各种类型的系统失效。在故障树术语中，这些是系统分析员可以考虑的“顶级事件”。分析员将选择其中一个顶级事件并调查其发生的**直接原因**。这些直接原因将是针对特定系统故障所选择的直接故障机制，并将构成某些子系统的故障。这些后一种故障将是子系统设计者的故障模式，将构成故障树的第二级。以这种“直接原因”的方式，一步一步地进行部件故障。这些成分是由树的分辨率极限定义的基本原因或所谓的“基本事件”。应注意识别其他系统中的故障可能导致部件故障的情况，并对这些故障进行调查。

从组件设计者的角度来看，树中较高的所有子系统和系统故障都表示故障影响；也就是说，它们表示特定组件故障的结果。组件设计者的故障模式是组件本身的故障。

如果组件设计者要构建一个故障树，那么这些组件中的任何一个故障都可能构成一个合适的顶层事件。换句话说，组件设计者的“系统”就是组件本身。设计人员的故障树的较低级别将包括组件故障的机制或原因。这些影响包括质量控制影响、环境影响等，在许多情况下，这些影响将低于系统设计者的故障树的分辨率限制，但仍需进行调查，因为它们可能是整个系统故障的主要来源。FTA之所以强大，是因为它超越了系统边界，深入研究了子系统或组件在总体设计中的作用

### 成功路径模型

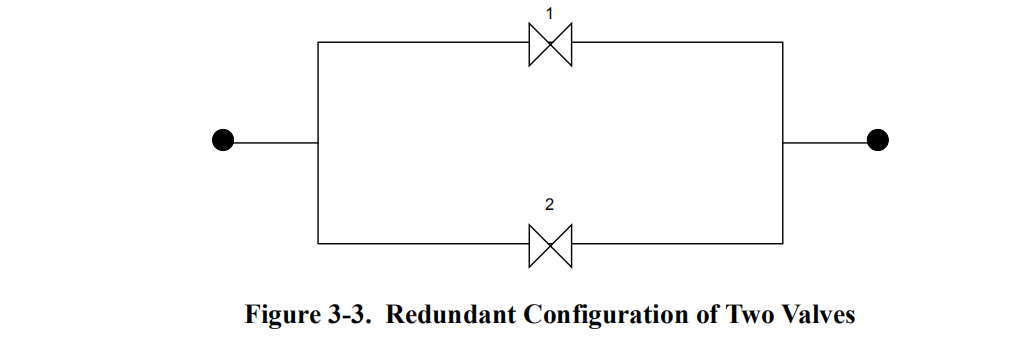
在大多数情况下，已经讨论了故障，故障仍然是本文件的主要焦点。然而，与其在“失败空间”中工作，不如在“成功空间”中等效地进行工作。在继续讨论失效空间方法之前，将提供一个等效性的简单示例。

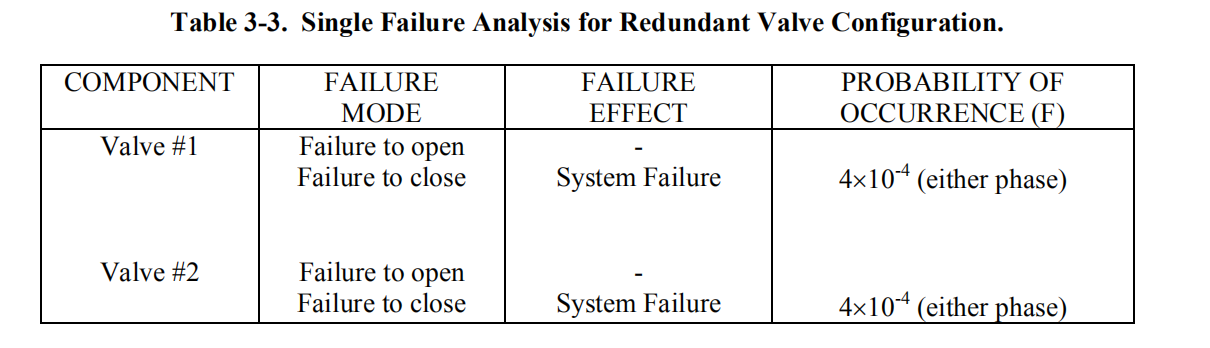
考虑图3-3所示的两个平行阀门的配置。可以通过考虑单个故障（假设多个故障的概率可以忽略不计）或考虑“成功路径”来分析该系统。首先考虑前一种情况。

系统要求如下：（1）操作分为两个阶段；（2）每个阶段至少有一个阀门打开；（3）每一阶段结束时，两个阀门都必须关闭。

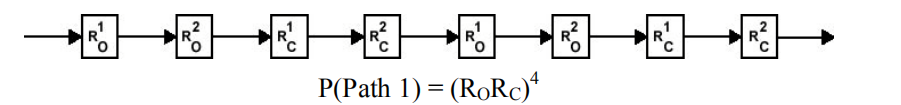
两种相关的部件故障模式是阀门无法按需打开和阀门无法按需关闭。为便于分析，假设每种失效模式的概率值如下：

P（阀门未打开）=1×10-4（任一相）

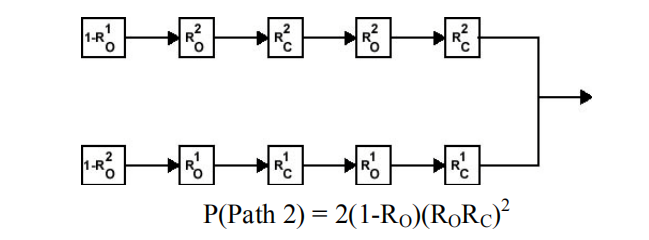
P（阀门不关闭）=2×10-4，其中符号“P”表示概率，假设阀门相同。

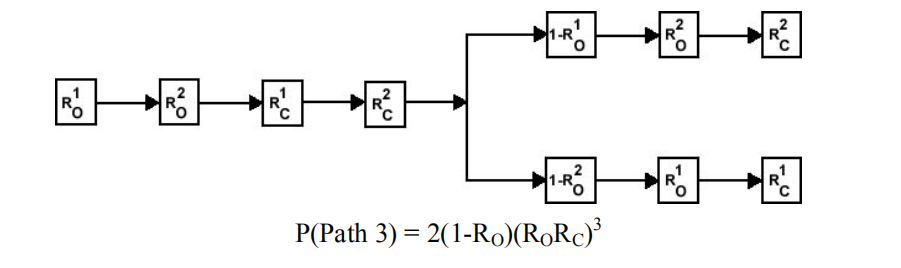
系统的单次故障分析可以如表3-3所示 

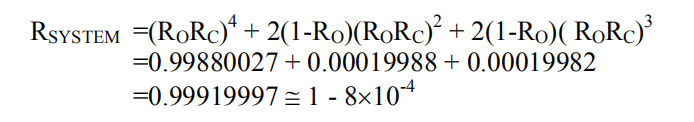
现在考虑是否可以通过考虑可能的成功来复制这个结果。有三条成功的道路，可以通过符号和示意图来确定。如果表示“阀i成功打开”，表示“阀i关闭成功”，并且P（路径i）表示与第i条成功路径相关的成功概率，可绘制以下图表：

路径1：两个阀门在两个循环中都正常工作

路径2：一个阀门在第一个循环中失效，但另一个阀门在两个循环中都能正常工作



路径3：一个阀在第二个循环中无法打开，但另一个阀在两个循环中都正常工作

从数值上讲，系统可靠性由

基本上和以前一样。然而，应注意到失效方法的难度要小得多

### 割集和路径集

当对故障树进行评估时，可以得到割集，也可以称为故障集。

割集是一组基本事件，如果它们全部发生，将导致故障树的顶层事件发生。由于基本事件是故障树中最底层的主要事件，割集将基本事件与顶层事件直接关联起来。最小割集（minimal cut set，简称minimal failure set）是一组最小的基本事件集，如果它们都发生，则会导致顶层事件发生。集合是最小的，因为如果没有发生任何事件，那么这个基本事件的组合就不会发生顶层事件。一个给定的故障树将有有限个唯一的最小割集。最小割集根据基本事件确定顶层事件发生的所有不同方式。一种获得最小割集的方法及其应用将在后面的章节中讨论。

割集的逻辑补码是成功树的路径集。一个路径集，也可以称为成功集，是一组事件，如果它们都不发生，将导致顶部事件不发生。因此，路径集将基本事件的不发生与顶部事件的不发生相关联。最小路径集（minimal path set，简称minimal success set），是指事件数量最少的集合，如果事件不发生，将导致顶层事件不发生。

集合是最小的，因为如果任何事件发生，那么这组事件不能保证顶级事件的不发生。成功树的最小路径集确定了确保顶级事件不会发生的所有独特方式。

## 故障树模型

### 模型符号解析

### 故障分类：主要、次要、命令

对于故障树分析人员来说，将故障分为三类：主要故障、次要故障和命令故障。主要故障是指部件在合格环境中发生的任何故障；例如，设计用于承受高达或包括压力PO的压力的压力罐，由于有缺陷的焊接而在某个压力p≤PO时破裂。

次要故障是指组件在未经鉴定的环境中发生的任何故障。换言之，部件在超出其设计条件的情况下发生故障；例如，设计用于承受高达或包括压力PO的压力的压力罐在压力p>PO下破裂。其他类型的次级故障包括微流星体撞击造成的损坏、环境控制系统丧失或过度振动引起的故障。

因为一次和二次故障通常是部件故障，所以它们通常称为一次故障和二次故障。相反，指令故障涉及部件的正确操作，但在错误的时间或错误的位置；例如，由于来自某个上游设备的过早或错误的信号，弹头中的防护装置过早关闭。另一种类型的命令故障是由于信号或控制电源丢失而导致部件无法运行

通常，分析师在构造FT时不会考虑这些故障类别。

例如，在只考虑主要失效和设计条件的默认假设下，可能不包括二次失效。然而，有许多构建的FT由于短途思维而不经意地省略了其中的一个或多个错误。即使分析员没有明确使用这些类别，它们也可以作为一个有用的检查表来确保

FT对不同类型的断层的覆盖是完整的。

### 被动/主动 元素

在大多数情况下，将部件分为两种类型，被动和主动（也称为准静态和动态）。被动式组件以或多或少静态的方式对系统的功能做出贡献。这样的部件可以作为能量从一个地方传递到另一个地方（例如，传输电流的电线或母线或传输热能的蒸汽管道），也可以作为负载的发送器（例如，结构构件）。为了评估非能动部件的运行情况，应进行应力分析、传热研究等试验。被动部件的其他示例包括管道、轴承、轴颈和焊缝。

活动组件通过以某种方式修改系统行为，以更动态的方式为其父系统的功能做出贡献。通常，这样的元件需要输入信号或触发器作为其输出信号。在这种情况下，有源元件起着“传递函数”的作用，这一术语在电学和数学研究中被广泛使用。如果有源部件发生故障，可能没有输出信号或输出信号不正确。例如，打开和关闭的阀门可以改变系统的流体流动，开关对电路中的电流也有类似的影响。为了评估有源元件的工作情况，对工作特性进行参数研究，并对功能相互关系进行研究。

一个无源元件可以被认为是一个“信号”的发送器。这个“信号”的物理性质可能有很大的变化，例如，它可以是电流或力。无源元件也可以被认为是“机构”（例如，管道），其中一个有源元件的输出成为第二个有源元件的输入。无源元件的故障将导致其“信号”的传输丢失

作为一个例子，考虑一个邮递员（被动组件），他将信号（字母）从一个主动组件（发送方）发送到另一个（接收方）。然后，接收器将以某种方式响应（提供输出），作为已接收到的消息（输入）。

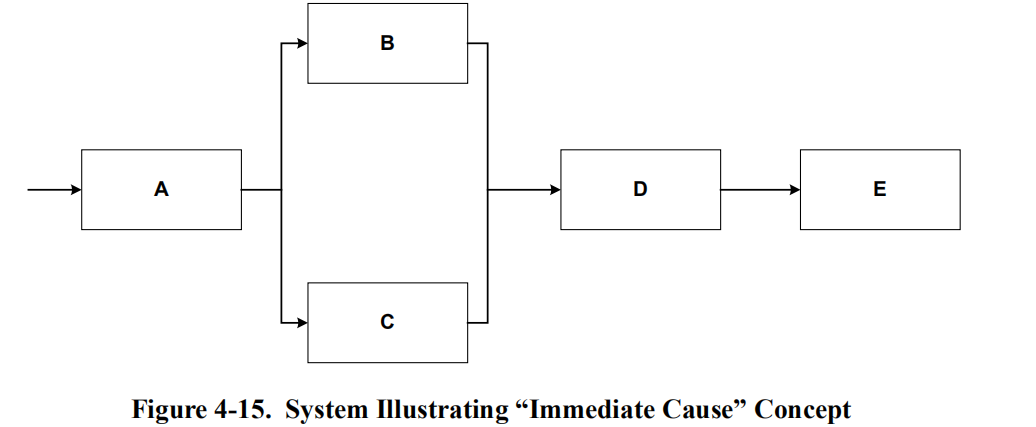
从数值可靠性的观点来看，有源元件失效和无源元件失效之间的一个重要区别就是失效率的不同。这两种类型的部件在可靠性上的差异通常是2到3个数量级。

在上述讨论中，主动和被动部件的定义适用于部件执行的主要功能：与该主要功能失效相关的部件故障。（例如，如果我们试图根据“主动”或“被动”定义对特定故障模式进行分类，则可以考虑主动部件的“被动”故障模式，例如阀门破裂。）

### “直接原因”概念

回到系统分析员的角度，首先定义所研究的系统（确定其边界），然后选择特定的系统故障模式进行进一步分析。后者构成故障树的顶层事件。接下来，分析员将确定导致这一重大事件发生的直接、必要和充分的原因。应该指出的是，这些不是事件的基本原因，而是事件的直接原因或直接机制。这是一个非常重要的问题，将在后面的示例中加以澄清和说明。

顶级事件的直接、必要和充分原因现在被视为次顶级事件，然后分析员继续确定它们的直接、必要和充分原因。在这样做时，分析员被置于子系统分析员的位置，对于子系统分析员来说，故障机制就是故障模式；也就是说，子顶层事件对应于子系统故障树中的顶层事件。

这样，分析者沿着树不断地将观点从一个机制转移到另一个模式，并在定义机制和模式时不断接近更精细的分辨率，直到最终达到树的分辨率极限。此限制由一种或另一种类型的基本组件故障组成，树现在已经完成

该系统的工作方式如下：发送给a的信号触发a的输出，a向B和C提供输入。B和C随后将信号传递给D，D最终将信号传递给E。a、B、C和D是动态子系统。此外，子系统D需要来自B或C或两者的输入信号来触发其输出到E。因此，系统的这一部分有冗余。

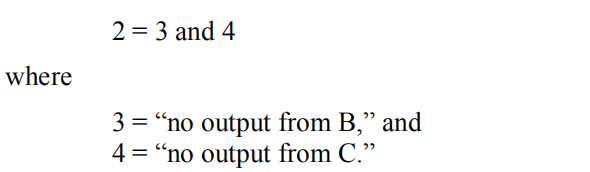
图4-13的系统可以很一般地解释。例如，它可以表示一个电气系统，其中子系统是模拟模块（例如比较器、放大器），也可以是管道系统，其中a、B、C和D是阀门；或者它可以代表公司“指挥链”的一部分。

将可能的结果“无信号发送至E”视为顶层事件，并假设可以忽略将信号从一个子系统传递到另一个子系统的发射设备（无源元件）。这相当于为导线、管道或命令链接指定零故障概率。

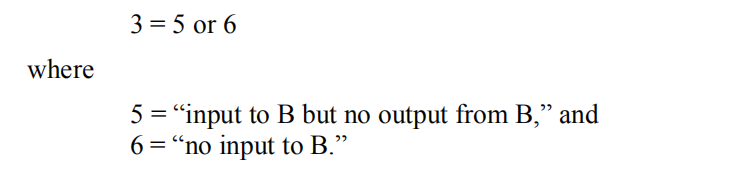
开始逐步分析顶级赛事。事件的直接原因“没有信号到E”，是“D没有输出”。分析员应强烈抵制将事件“无输入D”列为“没有信号到E”的直接原因的诱惑。在确定直接原因时，应一次采取一步行动。“直接原因”概念有时被称为“小思考”规则，因为这种方法是有条不紊、一步到位的。

现在，子顶级事件“无D输出”已经确定，接下来需要确定其直接原因。有两种可能：（1）“有一个输入到D，但没有从D输出。”（2）“没有输入到D。”

因此，事件1或2的并集可以产生子顶级事件“no output from D D”。（读者应该注意到，如果我们采取了不止一个步骤，并且（不正确地）确定了“没有输入到D”的原因，那么上面的事件1就会被忽略。事实上，考虑直接原因的动机现在很明确：它保证序列中没有故障事件被忽略。）

现在可以调查新模式故障的直接原因，即事件1和事件2。如果分辨率的限制是子系统级别，那么事件1（可以重新表述为“由于D内部的某个故障，D无法执行其适当的功能”）将不被进一步分析，并构成树的基本输入。对于事件2，其直接的、必要的和充分的原因是“B没有输出，C没有输出”，它表现为两个事件的交叉，即

就术语而言，如果进一步分析事件（如事件2），则将其称为“故障”是很方便的。然而，像1这样的事件代表一个基本的树输入并且没有被进一步分析被称为“失败”。这个术语也相当符合先前给出的“故障”和“故障”的机械定义。

继续进行分析，重点关注事件3和4。 就3而言

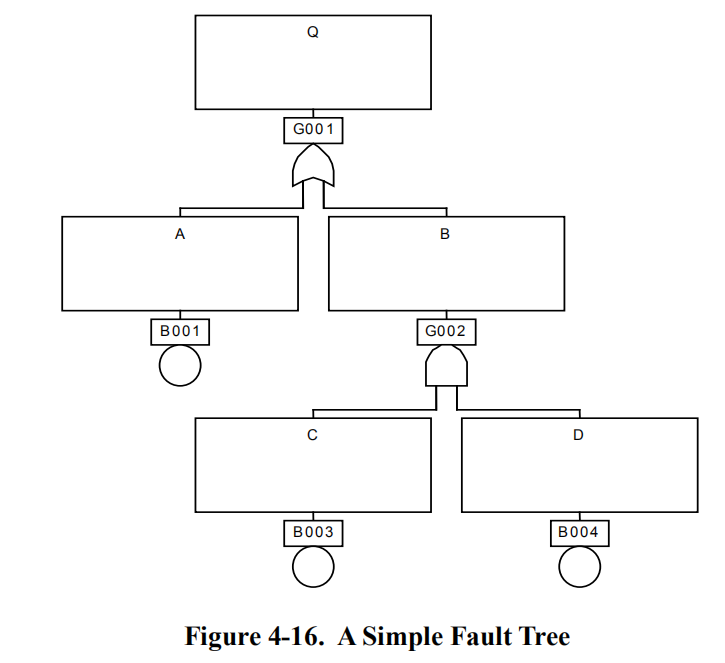
事件5可以很容易地识别为故障（基本树输入）。事件6是可以进一步分析的故障。事件4以类似的方式处理。

读者现在可以很容易地提供该系统分析的进一步步骤。当所有相关的基本树输入被识别后，分析将终止。在这方面，事件“no input to A”也被认为是一个基本的树输入。

我们对顶层事件（“无输入到E”）的分析结果产生了由“and”和“or”逻辑连接的故障事件的链接。“挂起”此链接的框架（或系统模型）是故障树。下一节将提供将故障事件链接连接到其框架（故障树）的必要细节

### 故障树构造的基本规则

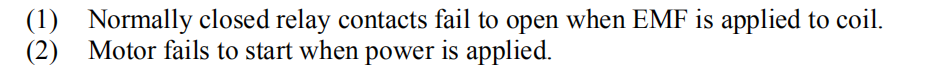
构造断层树是一个经过大约50年时间逐渐演变的过程。一开始它被认为是一门艺术，但很快人们就意识到，成功的树最好是按照一套基本规则画出来的。遵守这些规则有助于确保故障树的成功，从而使该过程不再是一门艺术，而是一门科学。现在将检查成功的故障树分析的基本规则。

考虑图4-16。它是一个简单的故障树，或者可能是更大故障树的一部分。请注意，没有一个故障事件是“写入”的；它们被指定为Q、A、B、C和D。

现在，当考虑一个特定的问题时，有必要准确地描述像Q，a，B，C，D这样的事件到底是什么。这样做的适当程序构成基本规则1：

**写下在事件框中作为故障输入的语句；精确地说明故障是什么以及故障发生的条件。不要把成功和错误混为一谈。**

“什么条件”描述了部件的相关故障（或运行）状态。“when condition”（当条件）描述系统相对于感兴趣部件的状况，该状态使部件的特定存在状态成为故障。

请注意，基本规则1可能经常需要相当冗长的语句。如果是这样，那就随它去吧。分析师被告诫不要害怕冗长的陈述。不要根据所画方框的大小调整语句的长度。如有必要，请将方框放大或缩小！（现代FT软件可能会限制语句的长度。然而，允许的描述长度通常足以充分描述一个事件。）允许缩写单词，但要抵制缩写思想的诱惑。故障陈述的例子有：

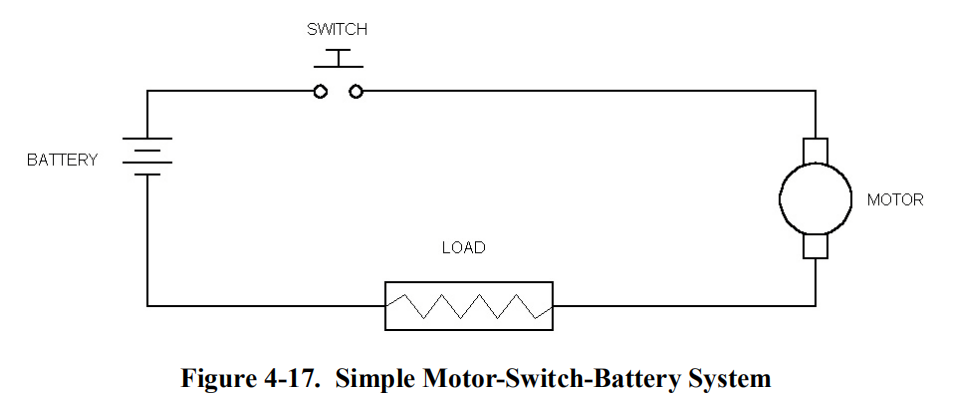
该过程的下一步是确定导致每个装箱语句描述的故障的必要和充分的事件。确定下一个要开发的逻辑结构的有用步骤是询问故障是“组件状态”故障还是“系统状态”故障。“部件状态”故障是指局限于给定部件的故障。“系统状态”故障是指不一定局限于给定部件的故障，但可能涉及系统级故障或多个故障的发生。

分析员不必使用“组件状态”与“系统状态”故障术语。然而，它提供了一个有用的检查表或助记符。重要的是质疑故障是否局限于一个部件，或者它是否可能涉及系统级故障或多个故障。

使用“组件状态”和“系统状态”作为助记符可以得出基本规则二：

**如果问题的答案是“此故障是部件故障吗？”？“是”，将事件归类为“组件故障状态”。如果答案为“否”，则将事件归类为“系统故障状态”。**

如果故障事件被归类为“组件状态”，则在事件下方添加一个或门，并查找主要、次要和命令故障模式。如果故障事件被归类为“系统状态”，则查找必要且充分的最小直接原因。“系统状态”故障事件可能需要一个AND门、一个OR门、一个抑制门，或者可能根本没有门。一般来说，当能量来源于部件外部的某个点时，该事件可归类为“系统状态”

为了说明基本原则II，考虑图4-17所示的简单电机开关电池电路

系统可以有两种状态：运行和备用。以下故障可使用基本规则II进行识别和分类：

