COSMIC 功能规模度量方法

版本号 **4.0.1**

实时软件的规模度量指南

版本号 **1.1**

2015 年 **4** 月

感谢

此指南文件(版本号 1.1) 的编者和评审者名单，此指南适用于 The COSMIC 方法 (版本 4.0、4.0.1)：		
Peter Fagg Pentad United Kingdom	Arlan Lesterhuis*, Netherlands	Hassan Soubra Ecole Supérieure des Techniques Aéronautiques et de Construction Automobile France
Charles Symons*, United Kingdom	Frank Vogelezang Ordina The Netherlands	Chris Woodward Chris Woodward Associates Ltd. United Kingdom

***** 这个指南的编者

指南文件(版本号 1.0，2012) 的编者和评审者名单：		
Alain Abran, École de Technologie Supérieure, Université du Québec, Canada	Juan J. Cuadrado-Gallego, University of Alcalá, Madrid, Spain	Jean-Marc Desharnais*, École de Technologie Supérieure, Université du Québec, Canada
Cigdem Gencel, Free University of Bozen/Bolzano, Italy	Arlan Lesterhuis*, Netherlands	Kenneth Lind, Viktoria University, Sweden
Bernard Londeix*, Telmaco Ltd, United Kingdom	François Perron, Pyxis Technologies, Canada	Charles Symons*, United Kingdom
Sylvie Trudel, Pyxis Technologies, Canada	Frank Vogelezang, Ordina, Netherlands	Steve Webb, independent consultant, United Kingdom

COSMIC 公开发行的文件和其他技术报告，包括其他语言的翻译，能通过 www.cosmic-sizing.org 的下载区获得。

版本控制

下表给出了这个文档版本的历史信息：

日期	评审者	修改/补充
2012 年 6 月	COSMIC 度量实践委员会	V1.0 版本发布。
2015 年 4 月	COSMIC 度量实践委员会	创建 V1.1 版本，以保持与度量手册 v4.0.1 的一致。

中文版序言

在 COSMIC 发布了针对实时软件的 1.1 版本的指南之后，我在过程改进之道的 QQ 群里（群号：133986886）发起了对其进行翻译的倡议，Emma(徐妍玲)积极响应，自告奋勇承担了对本指南的翻译。Emma 有着 16 年的软件研发与管理经验；拥有 PMP、CQM、Scrum Master 等相关职业资格；对 CMMI、ITIL、Six Sigma、People-CMM、ISO9000 等标准方法有较深入的理解，为不同企业提供过程与质量改进的全面解决方案。她对软件的量化管理一直充满高度热忱，并一直关注着“软件规模度量”的研究与实践。Emma 曾经参与过 COSMIC 的培训班，并顺利通过了 COSMIC 方法的认证考试，她很热心从事对 COSMIC 方法的推广工作。

在翻译时，我和 Emma 做了分工，她主译，我校对。我们力求既能忠实地表达原文的含义，又能通俗易懂。由于本指南专门针对实时软件，术语的专业性很强，很多案例中的用语也很专业，因此在翻译时需要查询很多资料，力求能够符合行业约定俗成的术语。

初期的翻译进展很顺利，Emma 利用自己的业余时间在比较短的时间内完成了本次翻译，而后期的校对我却由于其他种种事情，耽误了进展，10 月 1 假期期间，Emma 督促我完成了本次校对。尽管经过了反复推敲，由于行业知识的所限，仍然难免有疏漏之处，请各位读者不吝指正，以便于及时更新。对译文的任何疑问都可以邮件给我:renjialin@measures.net.cn。

我们希望 COSMIC 这种简单易行的规模度量方法能够在中国落地生根，得到广泛推广，给中国软件企业度量软件规模、估算软件开发成本带来一种标准化的、相对客观的方法。为了促进该方法的交流学习，我创建了 QQ 群：309842452，希望有兴趣的同仁参与进来共同提高。我们还会继续推出与 COSMIC 相关的其他指南的中文版，也希望大家能够积极参与翻译工作。

任甲林
COSMIC MPC 成员，IAC 成员
CMMI 主任评估师
麦哲思科技（北京）有限公司
2015 年 10 月 8 日

前言

本指南的目的、与 COSMIC 度量手册的关系：

这个指南的目的是为了帮助那些工作在实时软件领域工作的人员实现：将他们在进行实时软件的需求确定和需求建模时常用的概念映射到度量软件规模的 COSMIC 方法的概念中。这个指南也提供了很多度量实例和典型案例。

因此，本指南也是个辅助工具，它帮助实时软件系统从业人员将常用的术语翻译到 COSMIC 方法的术语。将 COSMIC 方法应用到实时领域时，除了在 COSMIC 度量手册[1].中提及的原则、规则外，没有其他额外的要求。

本指南的目标读者

本指南旨在提供给那些参与实时领域软件产品的定义、规格化、开发和管理的工作者。包括负责使用 COSMIC 方法来度量实时软件功能规模的软件度量小组成员和(或)开发人员，也包括那些对理解和使用这些度量结果用于项目绩效度量管理、软件合同控制、项目工作量估计等感兴趣的人员。这个指南并没有与任何特殊实时软件开发方法或生命周期模型捆绑，尽管可能在后面案例中会看到一些具体的实施需求确定或建模方法。注意：COSMIC 并不推荐任何特定的方法或工具。

这个指南的读者被假定为熟悉 COSMIC 度量手册[1].的人员。为了维护方便，本指南与度量手册之间具有很少的重复内容。

本指南应用范围

本指南关注“实时软件”的度量，这里的“实时软件”是广义泛指。按维基百科的定义，一个实时系统是指：一个操作的总的正确性不仅是取决于其逻辑的正确性，还取决于它执行的时间。按错过截止时间的后果不同，实时系统被分为“硬实时、准实时、软实时”[1]。

从本指南的目的出发，我们将那些由时钟机制控制的操作软件也包括在内。COSMIC 方法能用于度量所有这些不同类型“实时”软件的功能。(但是，要特别指出的：一个特定时间约束、或“时间期限”，比如：“所有命令必须在 1 毫秒内响应完成”是一个非功能性需求。COSMIC 功能规模方法是度量用于实现这些约束所需要的任何功能；但是，这些约束的数据值本身(1 毫秒、1 微秒或是其他任何)实际上并不会影响软件功能的规模。)

实时软件的例子有：工业系统的监控、从环境和科学实验中自动的采集数据、车辆系统的监控如发动机、通风设备、防撞系统以及家用电器上的应用。大的方面，实时系统控制着全球的电话网络、个人飞机和空中交通、发电厂等。一些软件系统，比如：酒店或航空订票系统，可能被描述成为商业应用系统和实时软件的混合体，因为

[1] 译者注：在维基百科中对“硬实时，准实时，软实时”进行解释的原文如下：

Hard – missing a deadline is a total system failure.

Firm – infrequent deadline misses are tolerable, but may degrade the system's quality of service. The usefulness of a result is zero after its deadline.

Soft – the usefulness of a result degrades after its deadline, thereby degrading the system's quality of service.

他们必须处理在实时约束下的查阅和预定操作。最后是中间件和基础支撑软件，比如：为实时应用程序提供基础任务和服务的操作系统，进而他们也在实时的约束下运作。

本指南内容的介绍

第一章讨论了：实时软件系统的特点、需求描述的方式和如何将它们映射到 COSMIC 方法的概念。第二章论述了：度量策略，特别是识别被度量软件的功能用户。第三章讨论了映射和度量阶段。第四章介绍了一些典型案例。

对于 COSMIC 方法中一般术语的定义，请参考度量手册[1]中的术语表。对于实时软件领域的专用术语，定义在本指南末尾的专用术语表中。要注意：一些术语被用于信息技术文献中有多种不同的含义，在 COSMIC 方法它们有特殊含义。因此，在使用本指南时，度量者要特别小心、正确使用这些 COSMIC 方法中的术语。

本指南版本是 v1.1，与 2012 年 6 月发布的 v1.0 版本相比，主要的不同在于：一些关于如何应用 COSMIC 方法的描述，以保持与 v4.0 和 v4.0.1 的方法一致性，包括一些编辑改进和对 4.7 节中添加了两个实例。此外，也纠正了 2.2 节下第三个例子中一些错误。

所有的重大变更的记录表，请查阅附录 A。

目录

1

将实时系统软件的需求映射到 COSMIC 概念

本章的目的是对实时软件领域下用于表达实时系统需求的各种方法，完成向 COSMIC 功能规模度量方法的概念和术语的映射 。如果功能规模的度量是针对某些现存的可运行的软件成品，度量者必须能使用 COSMIC 术语，对通过逆向工程得到原始的功能用户需求(FUR)执行同样的概念映射以完成规模度量。

回顾 COSMIC 方法的关键特征，一次度量必须执行三个阶段，见[1]:

在度量策略阶段，目的是确定度量目的，进而确定被度量软件的范围和功能用户，如：发送或接受数据到或从被度量软件处的人或事物。

被度量的软件的需求颗粒度等级和分解层次也在这个阶段被确定。

在映射阶段，目的是映射软件的 FUR 到 COSMIC 方法的概念。[注意：从 COSMIC 方法 v4.0 版本开始，我们使用“FUR”代表那些只有被完整定义的功能用户需求，以实现一个精确的 COSMIC 功能规模度量的可能性。对于那些被定义在较高颗粒度级别的需求，换句话说，就是那些无法细化到 COSMIC 规模度量要求的颗粒度的需求定义，我们将视情况使用“需求”或“功能需求”来表示。]

映射阶段有两个主要步骤：

- 识别被功能用户检测到或是产生的、软件必须响应的“触发事件”，进而识别对应的“功能处理”（见: 3.1 章节）。
- 识别被度量的软件块所引用的“兴趣对象”和“数据组”，进而识别在每个功能处理中的“数据移动”（输入、输出、读和写）（见: 3.2 章节）。

在度量阶段（见:3.3 章节），一个软件块的功能规模，是通过遍历其所有功能处理下的数据移动的总数来度量。对需求变更的功能规模度量，是通过统计满足变更需求所必需的增加、修改或删除的数据移动个数而得到。

注意：无论什么时候我们提及“触发事件”、“功能处理”、“数据移动”等，都是指对应的类型数，而不是出现次数。（参见度量手册[1],1.3.3 章节）。

1.1 实时系统软件的特征

接下来的章节中，一些与软件功能规模度量相关的实时系统的特性将会被介绍。

1.1.1 事件驱动的系统

实时软件通常被刻画为“事件驱动”， 也就是说，它的功能是必须在实时约束下对事件做出响应。它的行为可以描述为有限状态机，软件必须要响应的时间可能影响软件状态。状态不一定必须改变，比如：一个查询功能处理的完成并不会改变机器的状态。

在 COSMIC 方法中一个关键的概念就是：被度量软件的功能用户感知到的或是产生的事件。功能用户通过一个输入移动数据组到一个功能处理来和软件沟通某个事件的一次发生，该输入触发了功能处理的执行。这一过程由下面的图展示，此图来自于 COSMIC 度量手册[1]。

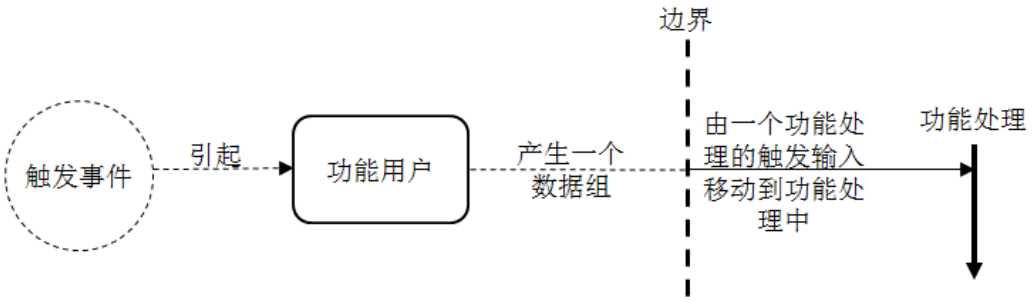


图 1.1 触发事件、功能用户及功能处理间的关系

三个触发事件的实例，用于解释图 1.1 的关系图：

实例 1：一个传感器检测到一个软件必须响应的信号

- *触发事件：被设计的传感器检测到的信号*
- *功能用户：传感器*
- *传感器产生和传送一个消息(一个数据组)，这个数据通过触发输入数据移动被移动到一个功能处理，通知说事件已经发生；这个消息也可能传输其他关于触发事件的信息。*

实例 2：软件 A 的软件块向软件 B 的软件块传递一个申请，请求一个服务

- *触发事件：当软件 A 需要从软件 B 获得服务时，它通过产生一个服务申请来产生一个触发事件（一个数据组提供了这个服务需要的输入数据）*
- *功能用户：软件 A 是软件 B 的功能用户*
- *服务消息的申请通过它的触发输入，被移动到软件 B 的功能处理；然后功能处理可以提供对应的服务。*

实例 3：每次时钟滴答时，软件块必须执行一个控制过程

- *触发事件：被时钟的滴答有效产生的触发事件（一个数据组）*
- *功能用户：时钟就是这个软件的功能用户*
- *时钟的“滴答数据组”被移动到功能处理，通过它的触发输入启动这个任务。*

在上面三个实例中，为了响应对应的触发事件软件必须执行的“任务”（服务）就是“功能处理”，它是数据移动的结果，只有到完全执行并满足了响应触发事件的用户功能需求才算结束。有三点需要注意的：

- 每个功能处理都是与任何其他功能处理相独立的，但是可能存在多种不同基数（1:n,1:1,或者 n:1）的链式组合，如图 1.1 所示：触发事件——功能用户——数据组——功能处理。（见：功能手册[1]案例）。

- 也许不同功能处理可能仅有一种特定的发生顺序，但是这并不影响每个功能处理必须独立被触发的事实。比如：一个“停止处理”不可能发生在一个“启动处理”之前，但是每个处理都是被一个单独的事件所触发。
- 被度量的功能规模不应该考虑：a)具体的实时时间约束，比如：响应应该在 1 毫秒内完成，这种是非功能需求（见：4.2 章节对计时器功能的度量）；也不考虑 b)当软件在等待触发输入时那些被“常规化”触发事件驱动的功能处理和其他“异常”触发事件所引发的中断（见：下一章节 1.1.2）。

1.1.2 中断

当一个功能用户检测到一个要求改变当前系统的事件时[2]，一个中断消息被产生。如果一个功能处理已经被触发和启动执行还没有正常终止时，一个中断信息可能有一个或两个可能的结果。

- 如果中断是被度量软件范围外的中断处理程序所控制的，比如在被度量的软件的另外一层，那么根据被度量软件的功能处理的限定，则可以不必考虑这个中断可能发生的事实。
- 另外一种情况，中断可能被传递给正在执行的功能处理，作为对正常触发输入的额外输入。在收到中断输入后，功能处理的行为由它的功能用户需求确定。相对于正常执行，这个行为导致额外的数据移动，需要按正常规则被度量。

1.1.3 功能规模会随着实时系统软件的功能用户的不同而变化

度量一个实时软件的功能规模最常见的目的是为了度量项目绩效和(或是）估计项目工作量。对于最常见的度量目的和范围，实时软件的功能用户将被识别为是直接与被度量软件交互的“事物”。通常这些会是：

- 硬件输入设备（如：传感器、测量设备、时钟）；
- 硬件输出设备（如：致动器、显示器、通信线路）；
- 其他可以发送输入或接收输出的软件块或硬件块。

然而，度量目的也可能是度量实时系统被人类操作员能看到的功能的规模（如：一个流程控制系统的操作台所提供的功能；或者一个简单的复印机或移动电话的操作员界面所提供的功能。）

人与软件的非直接交互，人们仅能感知的那些通过输入-输出接口能使用的功能。这些功能可能比软件必须提供的完整功能要少很多。因此，当度量实时软件功能规模时，为了保证度量的有效性，清晰地定义功能用户非常重要。

更多关于功能用户类型的信息，参见 2.2 章节。

1.1.4 实时应用软件：嵌入式、在一个或多个联合的操作系统执行实时应用软件？

我们使用术语“应用软件”指基于干系人的要求而开发的、用以执行特定的一系列任务的软件；术语“基础软件”指用于支持应用软件的操作系统、软件数据程序或设备驱动程序。

[2] SO/IEC/IEEE 24975:2010 系统和软件工程--词汇表对“中断”的标准定义为：（1）一个处理的暂停(或终止），以处理外部来的事件；（2）导致一个处理的暂停（或终止）；（3）宽泛的指：一个中断申请。（“终止”已经在这个指南中被增加。）

实时应用软件可能嵌入在芯片中(片上系统 SoC)，比如：现场可编程门阵列（FPGA）或是可编程逻辑控制器(PLC)，它自身可能就是某种专门应用，如：适应于坚固耐用的运行场景。一个简单的嵌入式软件应用直接与各种硬件设备的输入与输出进行交互，这种情形也很常见，此时不需要操作系统。

另外，实时应用软件也可能被安装到一个通用或是特定的处理器，在实时操作系统（RTOS）支持下运行。实时操作系统（RTOS）可以处理传递到或来自于各种硬件输入、输出设备的信息；它位于软件架构中与应用层不同的其他层（基础层或更低层）。如果目的是为了度量应用软件，根据 COSMIC 方法的原则：度量范围必须限制在具体某一层，那么实时操作系统（RTOS）的存在或任何其他层的基础软件必须被忽略。此原则也同样适用于对任何位于基础软件层中的软件的度量，度量范围必须一直保持在同一层。

联合系统，又称“系统的系统”由上述各种类型的处理器的多种元素组合，通过共同总线或是网络进行通信。在系统中的每个元素都应该被单独度量。

1.2 需求的描述

1.2.1 将需求分配到硬件或软件的难题

度量实时软件功能规模的一个难题是：通常需求被描述在“系统”层面，比如：他们被分配到硬件或软件（比如：在片上系统概念中），而不是明确地在软件层面。显然，在确定和文档化哪些需求被分给软件之前，是很难对软件功能规模度量达成共识的，因为对于“软件将要做什么”存在着各种不同的假定。

COSMIC方法原则上能被应用到信息处理的功能需求，在他们被分配到软件或硬件前，无论实际最终分配的决策是怎样的。举例来说，在没有任何相关软件、硬件知识情况下，使用COSMIC方法去度量一个便携式计算器的功能规模也是容易的。但是，假如要运用COSMIC方法度量分配到硬件的功能用户需求规模，在判定这种做法是确实有效之前，需要在实践中进行更多的检验。

当系统需求如何被分配到软件和硬件并不清楚的情况下以及对需求分配的决策没有专家建议的情况下，度量者必须文档化所有关于分配需求的假设并度量软件部分的规模。同时，度量者也必须指出预期软件功能规模的上下限。

1.2.2 EARS 语法下的需求描述

一个众所周知的事实：使用自然语言描述的需求从本质上是不精确的。因此，有很多方法被开发来支持自然语言的结构化书写，以提升需求的清晰度。其中一个实例就是 EARS（需求语法的简单方法[4]）方法中通用的需求语法，完美符合 COSMIC 模型中对功能处理的识别。其中，一些 EARS 语法等同于 COSMIC 概念：

- 当<触发器>时，等同于“触发事件”
- <系统响应>，等同于“功能处理”
- 在<在某种具体状态>条件下，可能包含了要求一个“输入”数据移动或“输出/输入”数据移动进行状态检查，或者是一个读操作，如果是状态被一个以前的功能处理所持久存储。
- 如果<触发器>，等同于“触发事件”

- 在<包括的特征>属于某个复杂需求，等同于上面提到的等同条款中一个或多个。

1.2.3 在有限状态机下的需求描述

实时软件需求的文档化，有时会使用一种叫“有限状态机”（或者叫“有限自动机”）的模型。 当有限状态机被用于表述软件需求时，它能展示出：期望软件要处理的有限数量的各种状态，和软件从一个状态变成另一个状态(也可能是返回原来同样的状态)下对应的触发事件或条件。从一个状态变成另一个状态被称为“状态转移”。通常，通过状态转移图和（或）其他表格，有限状态机是可视化的。注意：一个触发事件可能对应一个或多个状态转移，这取决于事件发生时对应所处的状态。在第四章提供了一个对使用有限状态机描述需求度量的有效实例；图 4.5[3]展示了状态转移图的实例。

忽略用于文档化有限状态机的规范，识别独立的功能处理的任务只取决于识别独特的触发事件——那些可能引发一个或多个状态转移的、从外界环境（比如：在被度量软件范围之外的）中已经改变的或是必须现在改变的事物。

两个实例

实例一 在一个汽车收音机/CD 播放器上的“Next”按钮是控制设备软件的一个功能用户。按下“Next”按钮会根据设备是处于“收音机”或是在“CD 播放器”下做出不同的处理。当在“收音机”下，设备必须切换到下一个电台；而如果在“CD 播放器”下，是执行换成下一个 CD 轨道。控制软件有一个功能处理来处理这两种情形，通过外界按下“Next”按钮来触发这个功能处理。如果功能处理已经在启动执行中，那么它就需要去决定它当前的状态，它可能会要求：

- *一个输入数据移动，如果软件必须获取从一个硬件功能用户来的状态。（假定不需要被告知：什么数据要被发送。），*
- *或者一个读，如果当前状态从持久存储介质里可以获取的话。（这个状态可能已经被前面发生的功能处理写到持久存储介质。）*

对于“按下 Next 按钮”触发事件，当功能处理完成所有必须的执行后，这个功能处理就完成了，即：包括了对“收音机和 CD 状态下”的处理。

实例二 当一个电梯门正处于“打开”状态时，“关门”按钮被按下，这会启动电机来关闭电梯门。当电机在执行过程中，电梯门是处在“正在关闭”的状态。当门关闭后，状态将会变成“已关闭”状态。

假设软件被要求来控制这个过程。“关门”按钮、电梯门的电机、和“门已关闭”的传感器都是这个软件的功能用户。整个关门事件有两个功能处理。第一个功能处理应该是被“关门”按钮被按下的外界事件所触发，启动电梯门的电机。电梯门状态从“门打开”向“正在关闭”转移。第二个功能处理是由“门已关闭”的传感器所触发。这个处理使发动机停止；门的状态从“正在关闭”向“已关闭”转移。（显然，这个例子可以在实践中要复杂得多，所需要的完整的响应取决于与安全装置的交互，等等。）

[3] 译者注：此处原文为图 4.3.2，有误，应为图 4.5.

1.2.4 可编程逻辑控制器的需求

IEC 61131-3：2013 标准[5]对可编程逻辑控制器（PLCs）定义了一套统一的编程语言的语法和语义，包括了两个文本语言：指令表(IL)和结构化文本(ST)；和两个图形化语言：梯形图（LD）和功能块图（FBD）。行为的执行取决于条件：包含布尔代数的文本语言描述的各种组合。

除了这些而外，PC 功能有时也可以借助其他模型实现文档化，比如：有限状态机(FSM)和（或）决策表。

三种图形化语言已经被定义来改善需求的清晰度，每一种都有配套的图形化规约。尽管在这些语言与 COSMIC 概念之间的等同性已经被识别(见：下面案例中的表格 1.1)，度量者在将这些图形化语言概念转化到 COSMIC 概念时，必须非常小心：

- 上述语言允许更宽泛的颗粒度水平。举例来说，在一个图形中的元素，在某种场景下可能是代表一个功能处理；但在另外一种情况下，它可能代表一个功能处理下的一些数据运算（COSMIC 认为数据运算隶属于数据移动的子处理，数据运算是不被单独度量的）。
- 上述语言可能是组合的，比如一些实际情况下，一个语言通过使用另外一种语言的规范来表达。
- 确定功能处理是困难的，因为并不是总是那么清楚：一个状态转移是由一个功能用户，还是被软件自身所引起的。
- 有时，功能用户和条件在文本上并不是那么显著。

功能块图（FBD）	梯形图（LD）	顺序功能图(SFC)	对应的 COSMIC 概念
输入变量的左边文本，输出变量的右边文本（箭头）	逻辑检查器（触点）， 致动器（线圈）	步骤之间连线的右侧文本	功能用户或是来/去的数据移动
输入变量的左边文本，输出变量的右边文本（箭头）	-	转移	驱动事件或是数据运算
块	-	-	功能用户或软件块
功能（块数量）	整个梯形图，相邻阶梯数、阶梯	步骤	功能处理或软件块
输入和输出变量的上方文本（箭头）	阶梯	活动	功能用户或数据运算的数据移动

表 1.1-图形化语言概念与 COSMIC 概念的对应

1.2.5 专业工具下的需求

有许多软件工具用于支持需求的管理，比如：捕获需求、分析需求、需求建模，以及帮助需求向设计的演变。原则上在这些工具中通过捕获和结构化需求，使功能规模的度量变得更加容易。但是通常情况下，是按“自顶向下”的方式逐步捕获更低颗粒度的需求。如果在项目早期就需要规模度量，然而需求的颗粒度级别未达到

COMISC 规模度量精细度的要求，那么就使用 CSOMIC 近似方法（见：参考文献[6]的案例）。

无论在工具中需求是如何被捕获的，必须将工具的绘图规则与 COMISC 方法的概念相对应，以便识别出：功能用户、触发事件、功能处理和数据移动的各种类型。

一个对需求工程工具的近期调查[7] 指出：绝大多数工具开发商还没有开发出一个产品可以按某种标准方法，比如 COSMIC 方法，实现对软件规模的度量。但是，这些工具的用户，尤其是实时领域的用户，正开始尝试开发出他们自己的自动化流程用以从需求工程工具中获取对应内容后，自动实现 COSMIC 功能规模的度量 。参见[8], [9], [10], [22], [23]中来自于汽车领域的针对嵌入在电子控制单元中的软件的案例，参见 4.5 节以及[11], [12] 和 [21]。

1.2.6 UML 描述下的需求

研究表明统一建模语言(UML)与 COSMIC 方法之间有紧密关系。Lavazza 和其他人 [13]分享了一个对 UML 完整描述的需求运用 COSMIC 方法的实践案例。

从统一建模语言(UML)向 COSMIC 映射时，要特别注意：用例视图可能是以不同详细级别的某种颗粒度来绘制的。一个用例视图可能对应的是一些或是一个功能处理，或是一个功能处理的某部分。一旦正确的颗粒度级别确定,测量相应的消息序列图是非常简单的。

关于如何捕获度量 COSMIC 功能规模所需要的所有信息和如何用这些规模估算软件代码规模，在[8]和[9]中定义了一个基于 UML 构件的概要描述。此外，它也说明了在工具中如何实现构件图（用例视图的另一种替代）和 COSMIC 之间的映射以及 UML 下模型化的信息如何输入到工具中。工具通过给出一系列度量值自动化了代码规模估算模型的选择和代码规模的估算。

[19]提供了针对采用 UML 模型描述的新软件的需求和改善型功能的需求进行自动化度量 COSMIC 规模的另外一种方法。

1.2.7 非功能性需求

对非功能性需求[1]的标准观点是：他们“包括但是不限于”：

- 质量约束（比如：可用性、可靠性、效率和可移植性）；
- 组织约束（比如：操作环境，目标硬件和标准的一致性）；
- 环境约束（比如：互操作性、保密性、隐私和安全性）；
- 实现约束（比如：开发语言，交付工期）。

注意一些非功能性需求适用于硬件与软件，比如响应时间；而另一些几乎只完全适用于软件，比如“可移植性”。

非功能性是容易误解的术语。Al Sarayreh 和 Abran 研究了 ECSS(欧洲航天标准化合作组织)IEEE 的关于非功能性需求的标准，这些标准中描述了诸如：可维护性、可操作性、可用性、可移植性等。他们发现：大部分的需求都会演变成可分配给软件的功能需求，此时，他们的规模都可以用 COSMIC 方法度量，比如[14]。在[15]非功能性需求的调查中，Symons 得出结论：对那些按常规被认为是非功能性的需求中，很大比例的部分，应该更好的被描述为“准”非功能性需求。这是因为它们中大部分，在项目过程中，全部或是部分最终会演变成可被度量的功能需求。（对于那些最终没

有演变成功能需求的剩余的非功能性需求部分，要么是可以明确量化的要么是一些命名约束，例如：响应时间，可用性，硬件平台，命名接口等）。

当在讨论关键任务系统时，比如空中交通管制或实时金融交易系统，Butcher[16]描述到：这些系统在项目初期近一半的文档化需求都是关于非功能性需求的。然而，在项目进展中，它们中大部分就演变成了功能需求。因此，他更愿意使用“直接”和“间接”功能需求，来代替按“功能”与“非功能”需求的说法。

如果在项目生命初期就有度量需要，度量者应该注意这样一些非功能需求和约束：他们会演变成可分配给软件（除了明确的功能需求之外）的功能性需求；并且，在度量整体软件功能规模时，要注意将这部分非功能需求也包括在内。所有相关假设都应该被文档化。

关于非功能性需求的更多信息，请参见度量手册[1]和“关于软件项目性能测量、基准和估计时如何解释非功能性需求的指南”[24]。

2

度量策略阶段

在开始实际度量前，需要确定 COSMIC 功能规模度量的策略，这个策略必须考虑各种参数。在本章中将讨论这些参数：

- 度量目的与度量范围；
- 被度量软件的功能用户（被度量软件的数据的发送者或预期的接收者。）
- 被度量软件的功能需求的颗粒度级别和软件本身的分解层级

通常，确定策略所需要的工作量并不大，但记录这些参数有助于确保规模度量的结果总是能被可靠地解读，即：后续度量用户总是能进行 “苹果跟苹果”的同类比较。

为帮助确定度量策略，“度量策略模式”指南文件[20]，为各种不同类型的软件，提供了一套度量软件规模的标准参数集，亦被称为“度量策略模式”。

2.1 度量目的和范围

2.1.1 度量目的

度量目的定义了为什么要做本次度量和度量结果用来做什么。

2.1.2 度量范围

度量目的决定了度量范围（定义了被度量功能的范围）。一个特定的目的可能需要度量多个单独的软件，即可能会有多个测量范围。

被度量软件的范围必须限制在单一的软件层中。关于区分层的更多信息参看度量手册[1]。

2.2 识别功能用户

度量实时软件时，与被度量软件交互的功能用户(“数据的发送方和/或目标受众”)通常有：

- 时钟（参看本指南的术语表）
- 提供输入的传感器（比如：温度、压力、电压等），被轮询、或是通过中断、或周期性地发送他们的数据和/或状态。
- 接收输出的硬件设备（比如：阀门或电机致动器、开关、灯、加热器等）
- 能触发功能处理的硬件芯片 (比如：看门狗芯片)
- “哑”硬件内存器，如：一个只能对数据的请求做出反应的只读存储器(ROM)；
- 通信设备（比如：电话线、电脑端口、天线、喇叭、麦克风等）；
- 人机交互的硬件设备（比如：按钮、键盘和显示器等）；

- 其他与被度量软件之间进行数据交互（提供/接收）的软件块。

“环境图”展示了被度量软件与它的功能用户之间的交互。通过绘制环境图有助于区分功能用户：

- 触发事件源（和被触发输入移动的数据组）
- 其他输入数据来源（比如：可能为非触发输入的轮询提供数据组）
- 预期接收方,或数据的目的地（发送输出给它）

一些功能用户可能担当不止一种角色，比如：与被度量软件交互的其他软件块、智能硬件设备或是通信线路。

上述所有类型的功能用户可能与被度量的软件进行直接、或是间接的交互，比如：通过操作系统或简单的设备驱动软件。但是，这些“使能”软件的功能性应该被忽略（当然，不包括它就是度量的主题的情况。）

实时软件也可能是从“人类作为功能用户”的角度进行度量，人类作为功能用户与被度量的软件发生直接或是间接的交互，比如：被通知了紧急情况的操作员：启动或是终止系统、设置参数、监控操作性能等。

考虑一下这种情况：人类操作员按下了一个按钮。是将被按下的按钮作为与被度量软件直接互动的功能用户，还是将按下按钮的人作为与软件间接互动的功能用户？（这里是不同的事件，如果是按钮，事件是“我被按下了”。如果是人类操作者，那么事件也许会是某种紧急情况下他／她发出一个警告）。如何选择取决于度量的目的。功能用户的选择必须是二选一；将这些不同类型的功能用户（人，硬件设备或其他软件块）混合在一起度量，或是分别度量两个求和，都是没有意义的。下面举例说明这种有选择功能用户的情况。

例 1．4.1.1 节描述了一个由一个可编程逻辑控制器(PLC) 控制的工业过程。度量目的是度量使系统工作所需的所有嵌入式软件功能,而不仅仅是限定于人类操作员所看到的功能。这一过程是由操作员按下一个启动按钮。但在这个例子中,考虑到度量目的, 功能用户是启动按钮，而不是按下按钮启动过程的操作员。

例 2．移动电话(手机)的嵌入式软件，与几种类型的按钮、屏幕 (可能作为输入设备或是输出显示)、移动电话的电池、喇叭、天线等都有交互。作为移动电话人类用户只看到软件需要提供服务的一小部分功能。因此，取决于选择不同的功能用户，可度量两个功能规模。

在附录参考文献[17]中，Toivonen 度量了两个移动电话被人类可视的功能，目的是为了比较他们的“存储密度”（功能规模／内存量）。对于手机制造商来说，这个是很重要的经济度量元。但 Toivonen 度量出的软件规模数明显少于软件工程师为满足所有的硬件/软件的功能用户需求而提供的所有手机功能。如下例所示，确定功能用户取决于需求：

例 3．一个或是更多按钮（类型）？

考虑一个工厂，它的移动生产线可以通过按下按钮来停止；在整个生产线的不同位置都有这样的按钮。度量者该识别一个还是多个功能用户（类型）呢？答案是，视对“必须被度量的功能用户”的需求而定。这个例子的问题是：是否按下按钮会导致不同的触发事件和独立的功能处理。

a) 需求:任何操作者可以在紧急情况下按下一个按钮来停止生产线。当按钮被按下时,系统记录下按钮被按下、生产线被停止的时间。沿着生产线上的许多按钮都提供同样的作用。这种情况下，识别只有一个功能用户类型和一个功能处理类型。

b) 需求与场景 a)一样，但同时还有一个在主管办公室的按钮的需求，这个按钮是提供给主管，在工作日结束时停止生产线。如果这个按钮也是与场景 a)同样作用的，那么仍然只识别一个功能用户类型和一个功能处理类型。

c) 需求与场景 b) 一样，不过增加按钮可在任何时间终止生产线的功能，需求是：当主管办公室的按钮被按下 3 秒后，系统马上停止生产线，然后创建停止/启动事件日志。（三秒的时间约束是按钮自身要求。）那么我们就需要识别两个功能用户（在生产线上的任意终止的按钮，和主管办公室的停止按钮）和两个功能处理。这两个功能处理共享一些功能（终止生产线），但是有不同的触发事件驱动（突发停止和工作日结束时）并且有不同的作用影响。

d) 需求与场景 c）相似，但是有一个额外的需求就是：主管有第二个按钮，用于启动或是重启被停止的生产线，并记录启动时间。现在，我们就有三个功能用户类型（生产线上的停止按钮，主管的终止和启动按钮），和三个功能处理（停止生产线，停止生产线并生成主管第一次按键的报告，在主管第二个按钮下启动或是重启生产线）。

参看度量手册[1]中 2.3 节和 3.5.9 节下实时样例 1。也可查看本指南 4.5 节下轮胎压力监测系统的样例。

2.3 识别颗粒度级别的分解层级

在开始某次需求度量之前，必须考虑以下需求工件的两个方面，以确保度量满足其目的，达到需要的精度，并能被未来用户准确的解释：

- 软件的“分解层次”。这个指将软件分解为可以交换或共享数据的独立组件。分解的过程可能会在需求定义阶段，当初始需求被认为太大不能由一个团队应付时，决定把系统分成不同的子系统，子子系统等，以在被分多次实现。
- 软件和/或软件组件的需求的颗粒级别 ，关注的是需求的详细程度。通常，在项目生命过程中，需求是以“自上向下”方式被生产出来，即，首先以大纲形式，然后随着项目进展制定出越来越多的细节(换句话说，以越来越低的颗粒度级别)。

  精确的 COSMIC 规模度量指的是：只有当细节足以识别软件必须响应的所有独立事件、功能处理和他们的数据移动。在还未获取这些信息时，如果需要规模度量，那么就使用 COSMIC 近似方法。包括了按需求颗粒度的级别的规模度量缩放到功能处理的颗粒度级别。在应用这些方法时，必须注意的是：在给定的时间点上，不同部分的需求可能是在不同细节水平。

请注意：在决定需求颗粒度级别之前，应先确定该软件的分解，因为颗粒度水平可能会由于不同组件而有所不同。在需求和软件设计演进中，都应该注意监控“分解层次”与“颗粒度级别”对度量方法的不同影响。

在考虑这 2 个因素时，实时软件没有什么特别的，更多细节可查阅度量手册[1]。“早期或快速 COSMIC 功能需求的规模度量指南”[6]中讨论了近似度量的几种方法，也提供了一个以更低颗粒度水平对一个电信软件系统的需求规模度量的实际有效案例。

3

映射和度量阶段

3.1 识别触发事件和功能处理

如 1.1.1 节中所描述的，软件是由功能用户世界的“事件”所触发。因此，识别触发事件是尤为重要的，它使度量者能识别出“功能处理”。在软件块的功能用户需求(FUR)中识别功能处理的步骤描述如下：

1. 从功能用户的世界，识别待度量软件需要响应的独立事件——“触发事件”（触发事件的识别可以通过状态图和实体生命周期图，因为某些状态转换和实体生命周期转换显示出软件必须响应的触发事件）；
2. 对软件响应的每个触发事件识别对应的功能用户；
3. 识别每个功能用户引发的用以响应事件的触发输入（也可能有其他的输入）；
4. 识别为每个触发输入所启动的功能处理。

案例：汽车里程表软件连接到位于驱动轴的转速传感器，测量其每分钟转数（RPM），还与钥匙插入传感器、时钟、和驾驶员的显示装置相连接。该软件的持久存储介质中包含了给预定义的各种显示装置要发送的消息的参数。在钥匙插入时里程表软件需要去获取显示参数，并初始化已安装的显示装置。时钟以 5 毫秒的间隔触发里程表软件从传动轴上获取转速信息，计算速度，并按适合于显示装置的参数发送速度信息以更新显示装置。

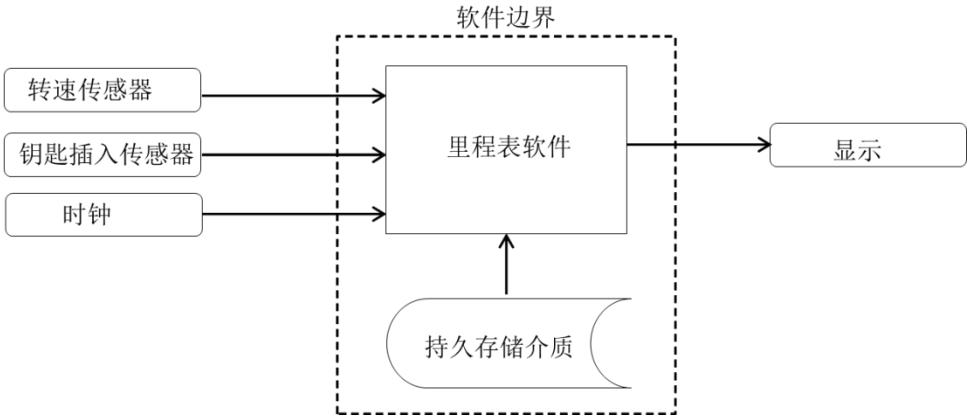


图 3.1 – 里程表软件的环境图

环境图显示了里程表软件的四个功能用户，即：三个输入设备（转速传感器、钥匙插入传感器和时钟）和一个输出设备。

有两个事件需要由里程表软件响应（即：触发事件），钥匙插入事件和 5 毫秒的时钟节拍。因此里程表控制软件有两个功能处理，FP1 和 FP2。

- *FP1 通过检测到“钥匙插入”事件初始化里程表控制软件，包括阅读显示设备的参数数据。*
- *基于时钟产生的每隔 5 毫秒的时钟节拍事件，FP2 测量速度并传递速度信息到显示装置。*

3.2 识别兴趣对象、数据组和数据移动

3.2.1 兴趣对象和数据组

任何功能处理都由称为“数据移动”的子处理组成，数据移动包括了移动数据，也被认为承担了相关的数据运算。一个数据移动移动了一个数据组，数据组的属性描述了单一的一个“兴趣对象”。

在实时软件中，一个数据组通常包括一个或几个数据属性，从输入装置如传感器发送给软件，或从软件发送到输出设备的一个信号，例如致动器。可以通过所涉及的设备来确定数据组所描述的兴趣对象。比如：在章节 3.1 中里程表控制软件的例子，“转速传感器”作为功能用户。这个传感器发送“当前转速”的数据组到软件。这个数据组的兴趣对象可以被认为是转动轴或是转速传感器。在实时软件中，一个功能用户（在上面例子中的转速传感器）也是被发送的数据组的兴趣对象（如：发送自己数据本身）。更多例子可参考度量手册[1]3.3.5 章节“功能用户作为兴趣对象”。

3.2.2 数据移动

有四种类型的数据移动：输入，输出，读和写。

输入和输出数据移动是跨越软件边界、分别从功能用户进来或是发出到功能用户的数据移动。读和写数据移动是与持久存储介质之间的数据移动。

对于实时软件而言，度量手册[1]3.5.9 章节的规则“当功能处理从功能用户处获取数据时”是尤其重要的。图 3.2 显示了在实时软件中从功能用户（具有不同的能力）、和持久存储介质接收或是得到数据的各种可能的方式。

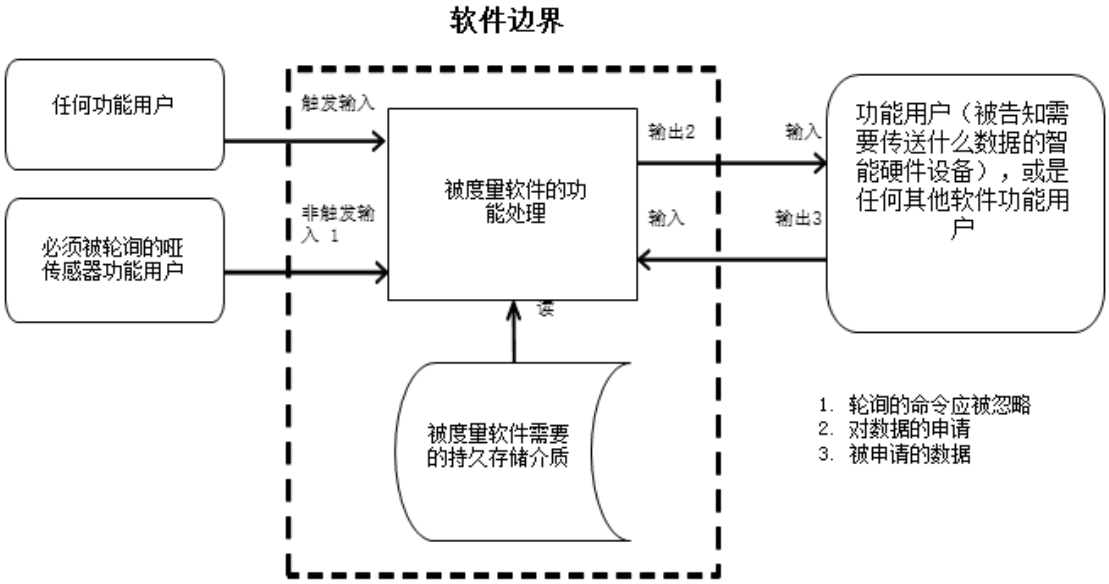


图 3.2 – 功能处理中接收或得到数据的各种方式

数据保存在存储介质中并可被读出：

- 通过另一个功能处理（数据的写移动）将数据存储在持久存储介质中，或是通过需要读数据的功能处理的前期执行，或是
- 在嵌入式软件的芯片被制造的过程中，被存储进物理只读存储器。嵌入式软件所需要的参数也可能以这种方式被存储。

对于一个单一字节的信号，比如时钟滴答，被作为输入数据移动（算成一个 COSMIC 功能点，即 CFP）可能会有一些疑问。但记住：每一个数据移动被假定是考虑了对应数据运算的，一个触发输入必须负责启动子处理，而不仅仅是一个字节的移动。

当然，在接收或是得到数据时，一个功能处理也可能是：

- 不需要响应的通过输出数据移动发送数据到功能用户
- 也能通过写的数据移动，“放”数据进持久存储介质。

3.2.3 数据运算

COSMIC 方法设计时并没有单独处理数据运算。如前面说的，方法假定数据运算是在每个数据移动中被考虑。

但是，这个方法也可能被合理的用于度量某些特定类型的需要大量数据运算的软件，参见度量手册[1]4.5.2 章节。这是可行的，例如：当软件必须处理大量数据，导致大量的数据移动类型。后者可以有效地解释有可能出现的任何数学复杂的数据处理。通过“合理的使用”，即针对各种度量目的，方法能产生出合理有用的规模，比如：项目绩效度量、估算、标杆对比等。如专家系统的规模、数字化处理连续变量的软件、从科学实验或工程度量中收集和分析数据的软件等。参见本指南的 4.7 章节的两个包括大量数据运算的软件度量的例子。

如果是对那些绝大部分都是“移动丰富”的，但也包括部分大量、本地化的数学算法的软件，COSMIC 方法允许做一些“本地扩展”。组织可以定义自己本地化的算

法规模的缩放比例和度量软件功能规模的 CFP 的缩放比例。另外，如果度量的目的是估算项目工作量，可将这部分算法排除在度量范围之外，这也是另外一个备选方案。规模和估算流程可以被仅应用于“移动丰富”的功能，对于开发算法的估算可以通过其他合适的流程来处理。关于该主题更多的内容，请查阅度量手册[1]的 “4.5.5 针对复杂算法的本地化扩展”内容。

3.2.4 在实时软件中的错误或故障信息

对于实时软件中包含的错误或故障指示的消息，必须以与其他数据移动相同的方式进行分析。

例 1：如果一个被度量软件发出的提示信息中，在一个特定兴趣对象的数据组中包括了故障或错误指示，作为所期望的数据的附加信息或替代了所期望的数据，期望的数据和此故障/错误指示描述的同一个兴趣对象，因此此时只有一个输出，即不要将故障/错误指示识别为一个单独输出。

例 2：如果故障原因或错误情况是被待度量软件发出的单独的信息，比如“错误源=传感器失效，内部错误”等，那么就需要被识别为一个单独的输出。

关于实时软件的错误信息的其他案例和定义信息，请查阅度量手册[1]第 3.5.11 章节中实时样例 1 和 2。

也可查阅本指南中第 4.1，4.2，4.3 和 4.5 章节，其中包括了将错误信息或故障情况作为正常输出被度量的案例。

3.3 度量和度量报告

关于以下内容的原则和规则请查阅读量手册[1]：

- 合并的度量结果
- 度量软件变更的规模
- 度量报告

所有这些主题都是领域独立的，即，不会因实时软件有什么特别的。

4

案例

4.1 工业自动化和可编程逻辑控制器(PLC)

4.1.1 可编程逻辑控制器(PLC)

可编程逻辑控制器（可编程控制器）是具有广泛的输入和输出设备，可以连接到传感器，致动器等类似的电脑。许多工业过程由 PLC 控制的，包括：传送带和相关机械、平板产品制造、流水线制造和化工过程。

4.1.2 化工厂过程控制的 PLC 软件度量

需求

一个化工厂由 PLC 控制的流程，有以下组成：用液体充灌，加热液体，然后当温度达到温度传感器装置的预设温度时，清空管内液体。

在下面的过程控制的需求描述中，我们假定：除非特殊说明，所有被提及的功能被分配到 PLC 软件。

- 过程启动：由操作员按下 PLC 控制所有后续步骤的启动按钮。
- 软件打开该罐的入口阀，并在重力下填充液体。
- 当罐充满（由高水位传感器检测到“到达高水位”），软件关闭入口阀，并启动加热器开始加热液体。
- 当软件接到已达到预设温度的通知，关闭加热器，打开出水阀启动罐排空。
- 泵持续放空，直到低水位传感器检测已 “达到低水位”，软件中止泵。
- 在整个过程中，过程状态（“充灌中”、“加热中”、“抽泵中”）都显示在操作显示器上。当过程完成时，会产生一个声音警报，“过程完成”的消息出现在显示器上。
- 当过程被启动和运行的过程中，PLC 软件都会轮询以请求阀门、加热器和泵的状态，定期检测任何故障条件。
- 如果 PLC 软件检测到一个错误，他会启动声音警报并向操作员显示对应设备的信息。如果操作员接收到一个错误消息，操作员会在软件之外通过手动处理。
- 轮询的频率取决于来自于一个时钟的信号。

环境图

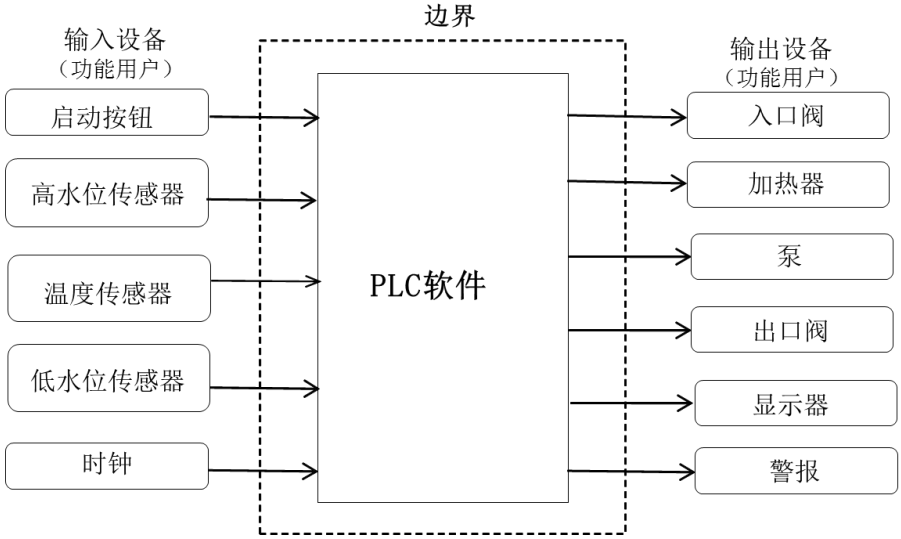


图 4.1 – 化工厂流程；PLC 软件和设备

分析

度量假定：所有与软件直接交互的硬件设备，都是功能用户，如上面环境图所示。PLC 没有操作系统。

如前面的需求描述，软件没有以任何方式分解，也不是其他软件块的构件。需求的颗粒度级别在“功能处理颗粒度级别”，即：单独的功能用户和事件的级别（而不是它们的群组）。

有各种触发事件（软件被要求需要响应的事件）和相应的功能处理（对触发事件的响应）如下表：

触发事件	启动功能处理的功能用户	功能处理的响应
启动按钮被按下	启动按钮	启动处理/充灌
达到高水位	高水位传感器	加热液体
达到预设的温度	温度传感器	停止加热/排空罐
到达低水位	低水位传感器	结束处理
时钟滴答（轮询时间）	时钟	失效检查

表 4.1 - 化工厂，触发事件和功能处理

被移动的每个数据组必须描述一个单一兴趣对象的某个方面。数据组包括了：从启动按钮，传感器和时钟传到软件的信息；也包括从软件发送到控制器、阀门和操作员设备的信号。

如 3.2.1 节中提到的，输入到软件的每个数据组的兴趣对象也是发出数据组的功能用户（即：功能用户是发送自身的数据）。类似地，每个离开软件的数据组的兴趣对象也是接收该数据组的功能用户（即：功能用户是被发送数据本身）。举例来说，

启动或停止泵的数据移动，移动了对应具体的泵状态的数据组。因此，泵是所有这些数据移动的兴趣对象。

对每个功能处理的触发输入，软件确定了要显示的处理状态（除了故障检查处理）。比如说：从启动按钮的信号，软件确定当前的状态是“充灌中”并显示这个状态。

这个 PLC 软件的功能处理如下所述，对每个功能处理提供：数据移动、被移动的数据组和解释说明。我们假定所有这些设备都是由软件轮询来判断其状态的“哑设备”，即：软件检测这些设备的状态，这就要求每个设备类型都针对每个轮询有且仅有一个输入（参见度量手册[1]3.5.9 章）。在下面功能处理中，“DM”代表“数据移动”。

功能处理:启动过程/充灌

数据移动	数据组	备注
E	启动信号	触发输入，从启动按钮
X	打开入口阀	给入口阀的信号，开始注入液体
X	给时钟的信号	激活时钟，按固定周期检测故障
X	显示的处理状态	显示“充灌中”

这个功能处理的规模是 4 个 CFP（COSMIC 功能点）。

功能处理:加热液体

数据移动	数据组	备注
E	达到高水位	触发输入，来自高水位传感器的信号，满
X	关闭入口阀	给入口阀的信号，停止注入液体
X	开启加热器	发送给加热器的信号，启动加热
X	显示的处理状态	显示“加热中”

这个功能处理的规模是 4 个 CFP（COSMIC 功能点）。

数据移动	数据组	备注
E	达到预设温度	触发输入，来自温度传感器的信号，被加
X	停止加热	给加热器的信号，停止加热
X	打开出口阀	打开出口阀的信号
X	打开泵	开泵的信号，启动排空罐
X	显示的处理状态	显示“抽泵中”

这个功能处理的规模是 5 个 CFP（COSMIC 功能点）。

功能处理:完成处理

数据移动	数据组	备注
E	到达低水位	触发输入，来自低水位传感器的信号，罐
X	停止抽泵	停止泵的信号
X	关闭出口阀	关闭出口阀的信号
X	显示的处理状态	显示“已结束”
X	声音警报	操作员的信号，处理完成
X	时钟信号	停止时钟

这个功能处理的规模是 6 个 CFP（COSMIC 功能点）。

（对轮询设备，“提示信息输入”被假定（参见度量手册 3.5.9 章节的规则 a）

功能处理:故障检查

数据移动	数据组	备注
E	启动信号	触发输入，来自时钟启动故障检查的信号
E	入口阀的状态	轮询入口阀的反馈结果信号
E	出口阀的状态	轮询出口阀的反馈结果信号
E	加热器的状态	轮询加热器的反馈结果信号
E	泵的状态	轮询泵的反馈结果信号
X	声音警报	如果检测到设备故障，给操作员的信号
X	显示的故障信息	显示故障设备

这个功能处理的规模是 7 个 CFP（COSMIC 功能点）。这个 PLC 软件的功能规模数总计为：4+4+5+6+7=26 个 CFP。

4.1.3 软件变更的度量

需求

已经决定去掉声音报警装置，并调整对应的软件。

分析

从软件到报警装置的信号可以被删除，即：在最后两个功能处理中，作为输出数据移动的声音报警数据组必须被去掉。变更的规模是 2 个 CFP。一旦变更执行后，软件功能的规模结果就是 24 个 CFP。

4.2 定时功能

（参阅：本指南的术语表，了解“时钟”和“定时器”的定义。）

对计时器功能的度量，要求明确地定义清楚：什么功能是被分配给硬件部分的功能，哪些是被明确分配给软件部分的。

案例 1. 需要的定时功能，如下所示，预设时间间隔的控制，可以通过在不同的软件或是硬件等不同的划分方式来实现：

- *硬件时钟按定义的固定时间间隔产生脉冲(“时钟滴答”)，每次都会触发一个软件的功能处理。软件保持对脉冲的跟踪，根据需要将其转换为秒或分钟，并积累逝去的时间直到到达设定的时间。*
- *硬件定时器包括产生并跟踪脉冲并将其转化为秒、分钟等，当需要时放在内部存储器中。软件启动计时器，当到达期望时间时，计时器通知软件。这种机制被用于下一个案例 2。*

案例 2. 一个 web 服务器必须访问客户信息系统来检索一些客户数据。除了处理这个请求，服务器启动一个监控程序来检查客户信息的请求被在设定的时间内处理。目的是确保查询客户信息的人类用户，不会因为客户系统没有反应，而陷入无限期的等待。图 4.2 采用消息序列图（没有重试）的方式显示了一个简单的案例，说明如何通过功能处理的四个参与者交互实现的，这些功能用户中每一个如下所述：

在向客户信息系统发出申请后，Web 服务器发出另一个消息对监控器，要求他报告是否超出了给定的时间期限。如果 web 服务器在规定时间范围内从客户信息系统接收到数据，它会告知监控器停止监听。否则，当 web 服务器第一次从监控器收到超时消息后，它就会发送给超时消息给申请客户信息的功能用户。

监控器记录从 web 服务器来的申请，向实时定时器发送一个申请，要求对方在规定超时周期内给出一个响应。（定时器可能在硬件或/和 RTOS 的软件内实现，这没关系的。）接下来监控器要么从 web 服务器接收消息后中止监听，或是从定时器接收到超时消息后完成。如果是后面这种情况，监控器会发送一个超时信息给 web 服务器。最终，监控器从它的日志中取消这个申请。

在图 4.2 中，定时功能性数据移动被显示为虚线。Web 服务器的申请监控功能需要 4 个 CFP，另外 2 个 CFP 来获取客户信息。监控器新需要 8 个 CFP 履行它的需求。（“删除申请”的写操作只被计算 1 次，尽管他可能在两种时机下都会发生，取决于客户信息是否在规定时间期限内返回。）

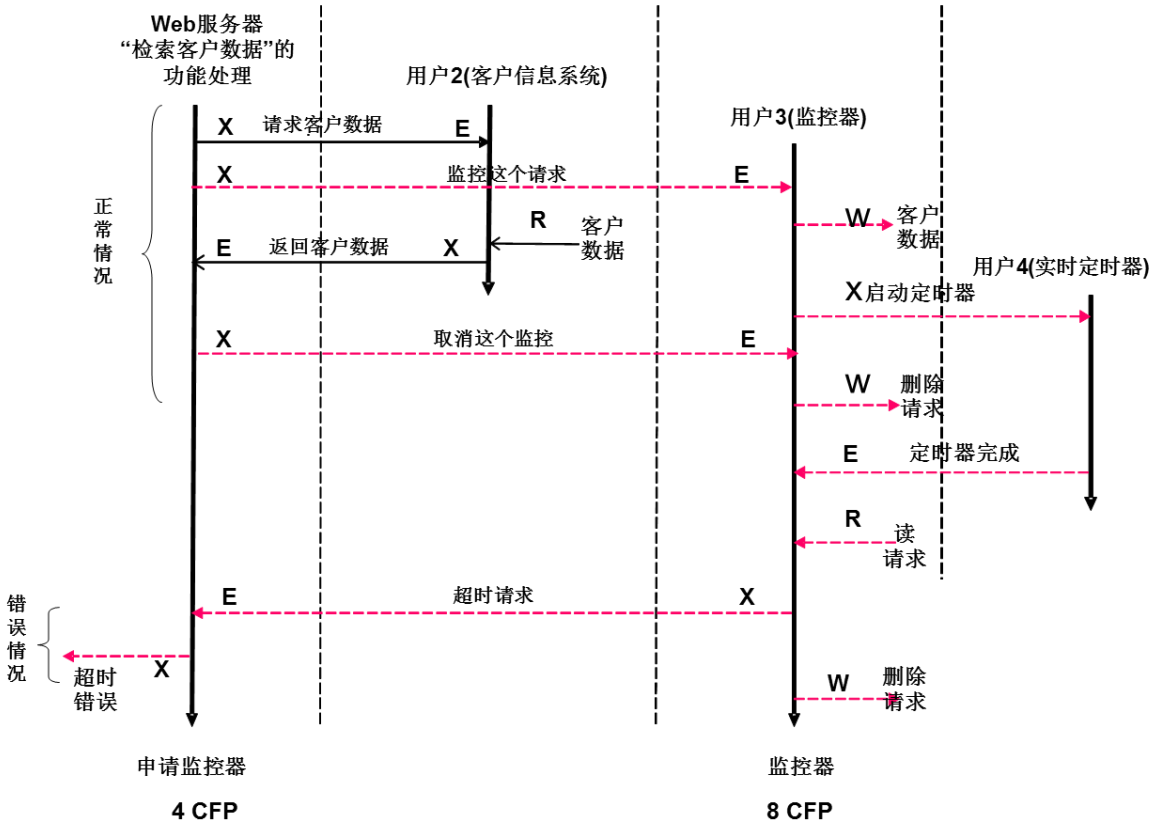


图- 4.2 web 服务器到监控器“超时”的功能

4.3 入侵报警系统

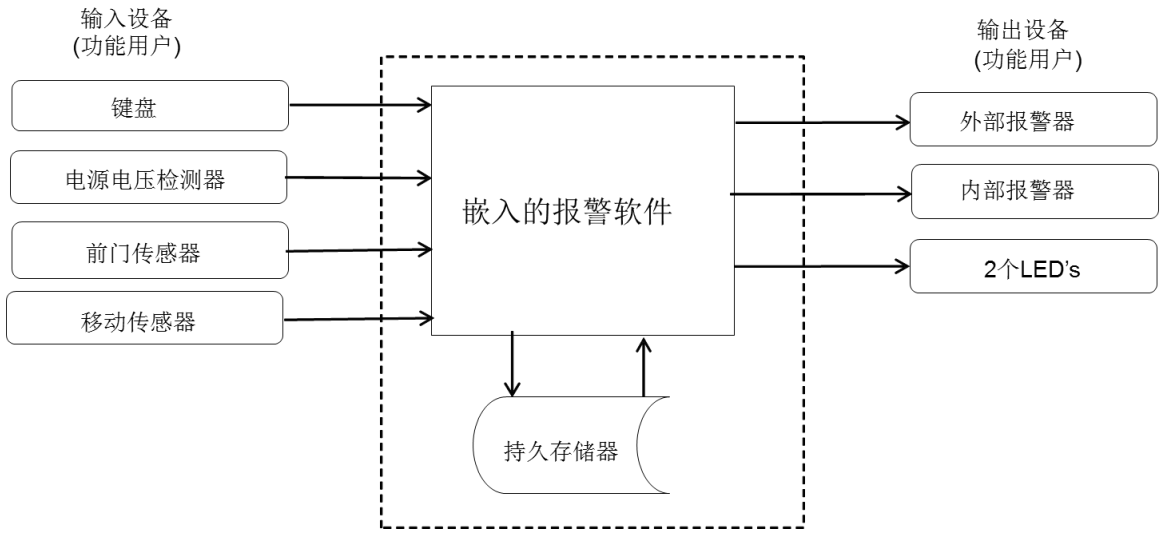
图 4.3 显示了一个关于基本的家庭入侵报警系统的嵌入式软件及其功能用户的环境图。

（假定的）需求:

(这些需求是从如何使用系统以及如何物理地测试系统推导出来的。我们只对正常居住者使用的功能感兴趣，不包括报警维护工程师的那些功能。)

- 软件连接到人机交互的键盘和红/绿 LED。
- 该软件也被连接到一个用于感应房子的主前门是否是打开的设备、几个内部的移动探测器，和一个内部和外部报警器。
- 当主电源发生故障时，还有一个备用电池接管，所以还必须有一个电源电压检测器。
- 最后，PIN(个人识别号码)代码被存储在软件中，还可以被修改，因此还需要一个持久存储器。
- 由于某些功能必须在预设时间内完成，因此还必须有一个定时机制。举例来说，已经在离开房子前设置了“退出代码”，则前门必须在预设的时间内关闭，否则就会警报。软件会在需要的时候启动定时器。我们不知道定时器是如何实现的，但似乎用硬件定时器是不可能的。因此，对于假定是由软件实现的定时器，对流逝时间跟踪的数据运算，是可以被忽略的。
- 对于外部警报器还有一个法律需求，就是必须在 20 分钟后停止下来。

图 4.3 入侵报警系统，环境图



分析

尽管输入源是来自人类操作员，输出也是显示给操作员，这又是一种 PLC 软件与其他各种输入输出设备之间的交互。我们不知道嵌入式报警软件是否包括一个操作系统，但存在与否，不会对这个报警应用软件的功能规模度量有影响。

下面是报警软件可能响应的一些单一类型的触发事件，每个都会触发一个单一的功能处理。可能还有更多事件，但是这些事件仅是针对维护工程师的，不算在度量范围内。

- 通过键盘输入新的或者被修改的 PIN 码 (2 个事件)。
- 通过键盘输入“退出代码”，从而激活系统(在离开房子之前和在预设时间内关闭前门)。
- 前门传感器检测到门已打开，同时激活报警系统。
- 系统被居住人激活或禁止，当他们在房间里时(2 个事件)，比如：在移动检测器范围之外，晚上休息时。
- 一个移动检测器发现一个移动信号，同时激活报警系统。
- 电源电压检测器发现电器失效或是复位的信号。（2 个事件）

作为一个例子，我们展示一下：前门从外面被打开，报警被激活的场景分析。当有人打开了前门，内部报警开始警笛声，PIN 密码必须在预设时间内被输入去禁止系统并停止内部报警的警笛声。如果没有在预设时间前没有输入 PIN 码，或者是输入错误的代码超过三次，启动外部报警轰鸣。

这里提到一个触发事件：“从外面打开前门”。作为功能用户的前门检测器，检测到事件并通知软件。如 3.2.1 节中提到的，在这种情况下，每个进入软件的数据组的兴趣对象也是发送数据组的功能用户（即：功能用户就是发送的数据本身）；类似的，每个离开软件的数据组的兴趣对象也是接受数据组的功能用户（即：功能用户就是被发送数据本身）。

假设功能处理是按如下方式工作：

功能处理: 响应从外部打开前门

数据移动	数据组	备注
E	“前门被打开”的信号	触发事件，传感器检测到前门被打开
R	从持久存储器中读取 PIN 信息	当前 PIN 码被存储作为一个持久数据组
-	启动内部定时器	我们假定这是个数据运算。被忽略
X	内部报警的信号	启动内部报警声
E	输入 PIN 码	代码验证是数据运算，附属于输入
X	软件处理 LED 红灯的状态从“打开”到“关闭”	PIN 码被正确、及时的输入。我们假定 2 个输出被要求。这个输出是关闭 LED 红灯

X	软件处理 LED 绿灯的状态从“关闭”到“打开”	PIN 码被正确、及时的输入。这个输出是点亮 LED 绿灯
X	停止内部报警声的信号	PIN 码被正确、及时的输入
-	再次输入 PIN 码，或是前次输入错误	这个前面输入的重复出现。这里是为了与规格说明书保持一致才列在这里，但应该忽略：重复出现的数据移动。
X	向外部报警声的信号	错误的 PIN 码被输入超过三次，和/或没有及时输入正确的代码，启动外部轰鸣报警
X	停止外部报警声	法律要求，在 20 分钟后

这个功能处理的规模是 9 个 CFP（COSMIC 功能点）。

讨论

- 是否要将“输入 PIN 码”的输入作为另一个触发事件？换句话说，这是一个功能处理还是应该将这个功能处理分解为两个？答案是“不”，“输入 PIN 码”并不是另一个触发事件。原因有二，首先，这个功能处理是被“前门打开”的消息所触发的。根据功能处理的定义：完整的功能处理必须满足功能用户的触发事件下所有可能的响应。第二，这个功能处理邀请（或驱动）了键盘功能用户输入 PIN 码。（键盘功能用户没有中断已经启动、请求允许输入 PIN 码的功能处理。)
- PIN 的输入可能被看作“四个单独数字的重复输入”，或是一个“四位数字”的输入。无论哪种方式，这里只有一个输入。（我们假设：在给定时间限制内检查是个数字输入是由硬件执行的。）。
- 注意这是一个从观察软件是如何工作的角度度量软件的例子，并不需求去理解详细的逻辑或精确的处理步骤。规模度量是考虑处理中各种可能的路径所引发的所有数据移动。这个表明对于所有对 COSMIC 概念有清晰理解的软件工程师，应该能对一个他们熟悉的软件系统实施度量过程。

4.4 在有限状态机下定义的电饭煲软件

需求

- 电饭煲软件能接收从它的门和启动按钮来的输入，也能发送信号去开关内部的灯和加热器。软件也能向定时器发送信号去设置烹饪时间，也能接收烹饪完成的信号。
- 当电饭煲的门被关闭，启动按钮按下，开始烹饪。
- 在烹饪期间门被打开时，关闭加热器。
- 在烹饪或是电饭煲门打开时，点亮电饭煲的灯!
- 烹饪时间是一分钟的倍数。
- 每按一次启动按钮，就增加一分钟的烹饪时间
- 当计时器停止时，要么因为门在烹饪过程中被打开，要么是因为烹饪完成的定时信号，定时器重置为零。

- 电饭煲软件的初始化超出了这个样例的范围。假设电源是打开的，电饭煲是在“待机”的状态。

环境图

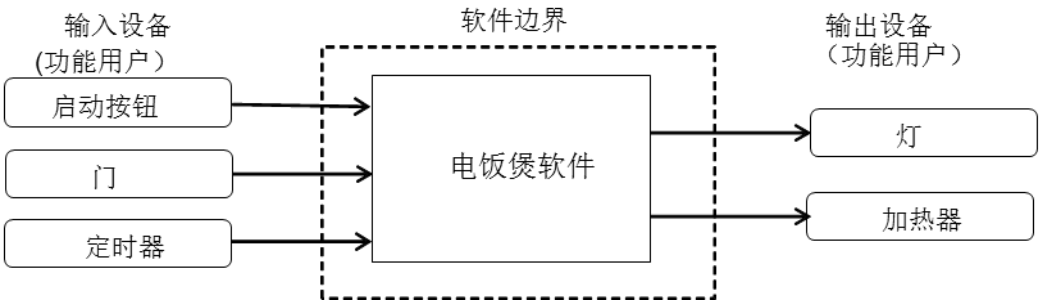


图 4.4 - 电饭煲，环境图

图 4.5 展示了电饭煲的状态转换图。盒子代表状态，箭头表示从一种状态转换到另一个(可能是相同的状态)。引发电饭煲在状态之间移动的事件是触发事件，用前缀为“TE”标识，而功能用户用“FU”作为前缀。

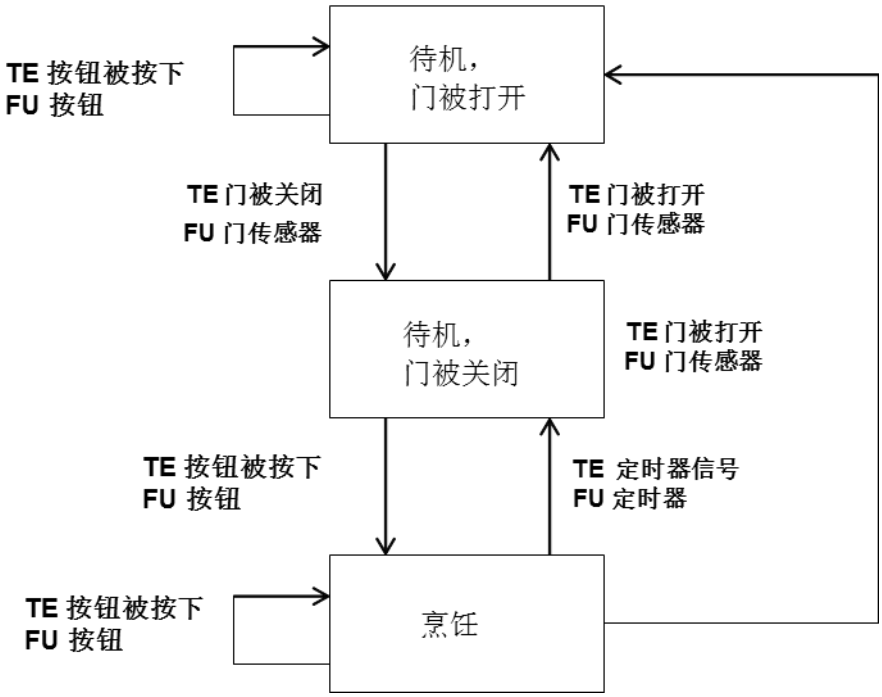


图 4.5 - 电饭煲，状态转换图

分析

输入端的电饭煲软件的功能用户是门传感器和按钮。在输出端的功能用户是电饭煲的灯和加热器。两边都有的功能用户是定时器。如 3.2.1 节中提到的，在这种情况下，每个进入软件的数据组的兴趣对象也是发送数据组的功能用户（即：功能用户就是发送的数据本身）；相似的，每个离开软件的数据组的兴趣对象也是接受数据组的功能用户（即：功能用户就是被发送数据本身）。

实际触发软件启动一个功能处理的事件如下所述，由于触发事件和功能处理是一一对应的，所以对他们使用了相同的名字。

表 4.2 – 电饭煲，触发时间和功能处理

触发事件	启动功能处理的功能用户	功能处理
门被关上	门传感器	门被关上
按钮被按下	按钮	按钮被按下
定时器信号（烹饪结	时钟	定时器信号（烹饪结束）
门被打开	门传感器	门被打开

电饭煲的功能处理如下所述：

功能处理: 门关闭

数据移动	数据组	备注
E	门被关闭的信号	触发输入，来自于门传感器
X	电饭煲的关闭信号	到电饭煲的灯

这个功能处理的规模是 2CFP（COSMIC 功能点）。

功能处理: 按下按钮

数据移动	数据组	备注
E	按钮信号	触发输入，来自按钮
E	请求门的状态	检验门是否关闭[4]
X	加热器信号	到加热器，如果门关闭了
X	点亮电饭煲灯的信号	到电饭煲的灯，如果门关闭了
X	启动或/和 按分钟增加烹饪时间	到定时器，如果门关闭了

这个功能处理的规摸是 5 CFP（COSMIC 功能点）。

功能处理: 时钟信号(烹饪结束)

数据移动	数据组	备注
E	定时器的信号	触发输入，来自：定时器
X	关闭加热器的信号	到加热器

[4]假定门感应器是“哑的”，一个输入就充分了。已经启动的功能处理，检测一个功能用户的状态并检索他要求的状态数据。参见度量手册第 3.5.9 节。

X	烹饪结束的信号	到电饭煲的灯

这个功能处理的规摸是 3 CFP （COSMIC 功能点）。

功能处理: 门被打开

数 据 移 动	数据组	备注
E	门被打开的信号	触发输入，来自门传感器
X	点亮电饭煲灯的信号	到电饭煲的灯
X	电饭煲灯熄灭的信号	到加热器
X	停止定时器	到定时器

这个功能处理的规摸是 4 CFP（COSMIC 功能点）。

度量范围内的电饭煲软件功能规模总计是：2+5+3+4 = 14 CFP。

讨论

在解释状态转换图时注意一个重要的问题：并不是所有的状态转换都对应到一个功能处理。在这个例子中有七次状态转换，但只有四个功能处理。只有检测到事件或由外部功能用户产生的到软件的事件，才可以触发一个功能处理。每个功能处理必须处理所有状态和状态组合以响应给定的触发事件。

举例来说，触发事件“按钮被按下”可能发生在电饭煲处于下面三种状态中任意一种。按钮被按下的事件发生在硬件的外部世界、是完全独立的机器状态。一个功能处理必须要根据“按钮被按下”时的机器状态，以三种方式处理“按钮被按下”的事件响应：

- 在“待机，门打开”状态，在已经发现门是开着的，它就停止；
- 在“待机，门被关闭”状态，它发送信号启动加热器、开灯、并启动定时一分钟的烹饪；
- 在“烹饪中”状态，它执行与前面状态一样的数据移动，但因为加热器已经启动、灯也亮着，结果就是增加一分钟的烹饪时间。

在这个例子中，我们假设电饭煲可以通过简单地检查门是打开还是关闭的，就能执行它的功能。在一个更复杂的情况下，软件可能需要记录机器的状态，并在状态改变时在持久存储器中进行更新。这将避免用软件来判断每次新事件发出信号时机器所处的状态。

类似地，“门打开了”事件可能发生在两种机器状态下。一个功能处理必须考虑如何处理这两种状态。

4.5 轮胎压力监测系统

需求

- 轮胎压力监测系统(TPMS)检测汽车的四个轮胎的压力。

- 每个轮子都有一个传感器，获取轮胎的压力。
- 一旦启动汽车电源，无论汽车是否开动，一个时钟会激活 TPMS 软件以每分钟一次的频率检索四个传感器的状态。传感器返回的轮胎的状态包括：传感器 ID（识别特定的轮胎)和轮胎压力。
- 如果压力过低或过高——软件中的数值——TPMS 打开仪表盘中的红色 LED 警报灯（对应传感器的位置）。
- 如果压力恢复正常，TPMS 关闭对应仪表盘的红色 LED 警报灯。
- TPMS 电子控制单元(ECU)、时钟、轮胎压力传感器和仪表板有 CAN-总线耦合在一起(CAN =控制器局域网总线)。

度量 TPMS 的目的是获得 TPMS 的功能规模。

环境图

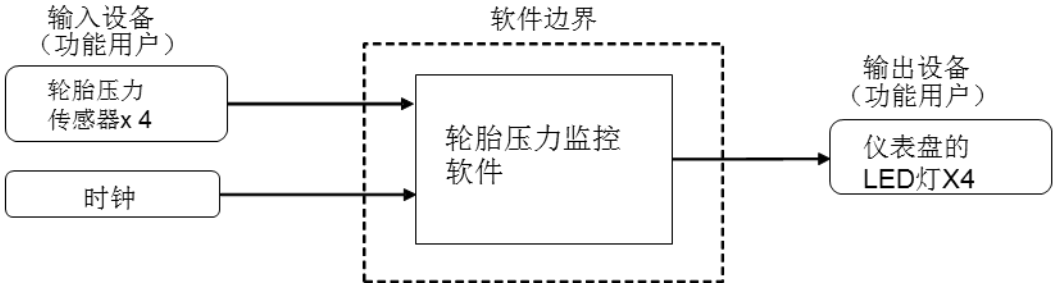


图 4.6- TPMS, 环境图

分析

在输入端 TPMS 软件的两个功能用户是时钟和四个轮胎传感器组。在输出端的功能用户是四个仪表盘的 LED 灯组，分别与汽车四个轮子相对应。

CAN-总线控制，组成了一个软件集合，将 TPMS 软件可以利用的服务耦合在一个软件层中，与其他 TPMS 软件所在层独立开来。因此，网络控制器不在这个度量范围内。注意：如果控制器在度量范围内，他们必须作为另外一层的软件，被单独度量。

软件必须响应一个触发事件，每 60 秒会有时钟信号发出，因此有一个功能处理：

表 4.3 – TPMS， 触发事件和功能处理

触发事件	启动功能处理的功能用户	功能处理
时钟信号	时钟	启动 TPMS 软件

如 3.2.1 节中提到的，输入到软件的每个数据组的兴趣对象也是发出数据组的功能用户（即：功能用户是发送自身的数据）。类似地，每个离开软件的数据组的兴趣对象也是接收该数据组的功能用户（即：功能用户是被发送数据本身）。数据组是：

- 时钟信号，兴趣对象为时钟

- 轮胎压力每次发生四组，每组数据都包括传感器 ID 和对应轮胎压力，兴趣对象是轮胎压力传感器
- 打开或是关闭一个或更多仪表盘上的 LED 报警灯的信号，兴趣对象是报警 LED 灯组

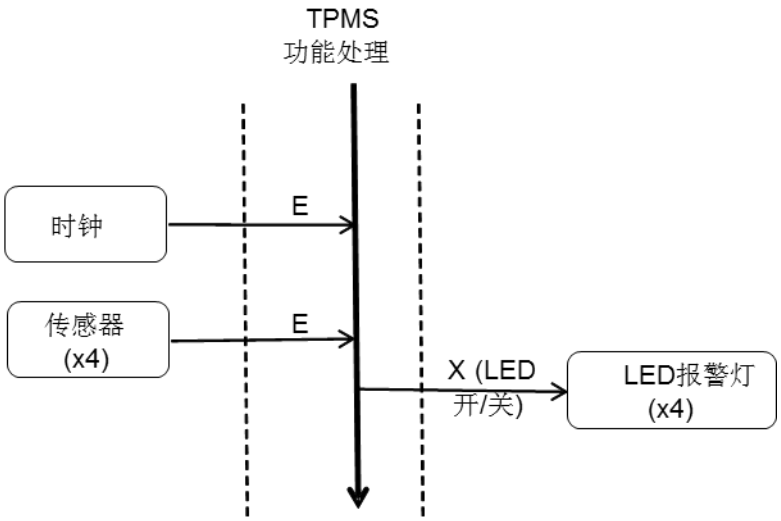


图 4.7 – TPMS 的一个功能处理的消息顺序图

该系统中没有要求任何持久数据的存储或检索。功能处理和数据移动如下所述。这个功能处理的数据移动、被移动的数据组和对应解释描述如下。

功能处理: 启动 TPMS 软件

数据移动	数据组	备注
E	启动信号	触发事件，来自于时钟
E	轮胎压力	获得四个“哑传感器”的数据，每个轮胎各一个
X	四个报警 LED 灯组的信号	必要时，点亮每个轮胎对应的 LED 灯。对压力数据的检查是数字运算，在输出数据移动中考虑。

这个功能处理的规模是 3 CFP （COSMIC 功能点）。

注意：尽管这四个“哑”传感器并不相同，他们必须单独被软件辨认。但是，对四个轮胎压力数据的要求是相同数据组的不同实例而已。因此，按照数据移动独特性规则（参见度量手册[1]），只能被识别成一个输入数据移动。被这个输入移动的数据组包含了 2 个属性——轮胎 ID 和对应轮胎的压力值。作为一个输入，也只涉及一个兴趣对象：轮胎压力传感器。这同样适用于输出，传递状态数据到四个 LED 灯。

这个简单的例子说明了许多实时系统非常重要的特征：有很多同样的传感器和输出设备的多次实例。比如可能是多种传感器 （除了分布位置不同外，其他都一样）分布在同一材料的一系列滚轴上，通过过程控制系统来控制。这些感应器（或是输出显

示设备）是相同的，所有输入（输出）数据移动的功能处理都是一样的，根据规模度量目的，计算的是数据移动的种类数，而不是发生的次数。

4.6 度量实时软件需求的自动化

在引用文件[25]和 Hassan Soubra[13]的博士论文中，提供了使用 Matlab Simulink 工具对汽车电子控制单元的实时嵌入式软件需求建模，实现自动化度量的例子说明。对此方法的一个简洁的英文描述，属于 Renault 版权，可以通过 www.cosmic-sizing.org 的下载部分获取。

引用文件[19]提供了对在 UML 描述下需求自动化度量。（没有特别说明是实时软件）。

4.7 数据操作丰富的实时软件的度量

在这个章节中的一个样例说明了这是假设的合理性：对实时软件的数据运算功能通过 COSMIC 方法来计算是合理的。当然，这个例子没有证明这个假设一直都是合理的。众所周知，功能需求的某些特定局部具有高强度的数据运算,请参考这个指南的 3.2.3 节。

某些航空软件中的算法分布

对非常复杂的航空系统下一个非常大的软件组件使用 COSMIC 方法进行度量。该组件的需求的总规模（存储在建模工具中）超过了 8000CFP。在 Ada 语言下需要实现的源代码行数超过了 8 万行。

此系统组件由 33 个子组件组成。在每一个子组件中，与每个数据移动关联的 Ada 代码行的也统计了，被称为“NOLA”，作为“算法的行数”。从而，可以为这 8000 多个数据移动计算“每个数据移动的 NOLA”。

图 4.8 展示了除了 5 个例外的数据移动以外所有的“数据移动的 NOLA”的频率分布直方图。五个例外的数据移动是：28(x2)，36，40 和 138 个 NOLA。(例如：直方图显示 8000 多个数据移动中有 20 个的 NOLA 为 7。)

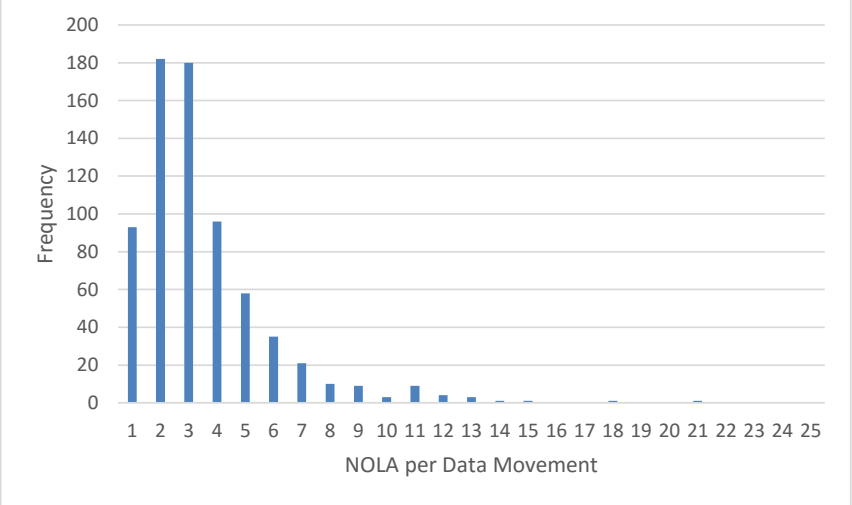


图 4.8 “每个数据移动的算法行数”的频率

从这些数据派生出来的参数：

Parameter 参数	Value 值

每个数据移动的 NOLA 的中位数	2.4
每个数据移动的 NOLA 的平均数	3.5
占 95%的数据移动控制上限	8 CFP
占 99%的数据移动的上限	14 CFP

数据和分析表明每个数据移动的 NOLA 值都有有限的范围，除了极少数的例外。这个发现，至少在这个特定的实时软件上，支持 COSMIC 方法关于数据移动能反应数据运算的数量进而能很好的反映功能规模的假设。

4.8 度量汽车电子控制单元的功能对内存空间需求的规模

由 C. Gencel, R. Heldal and K. Lind 的一篇名为“在生命周期里的软件产品的规模转换”的论文，在 2010 年 11 月斯图加特的软件度量国际研讨会上，提出了应用 COSMIC 方法度量瑞典生产的萨博汽车中嵌入电子控制单位的软件规模。这项研究的目的是检查基于 COSMIC 度量的功能规模和由此产生的以字节度量的目标代码所需的内存空间之间的关系。他发现了一个非常好的线性相关性。

在此论文中，作者陈述到：“本文表明可以获得准确的代码大小估计即使软件组件包含复杂的计算，只要组件包含相似的复杂性与组件接口的数量成正比。”

Renault [26]也指出： 以代码行规模与基于 COSMIC 度量的 CFP 功能规模的有很强的相关性，如下图所示：

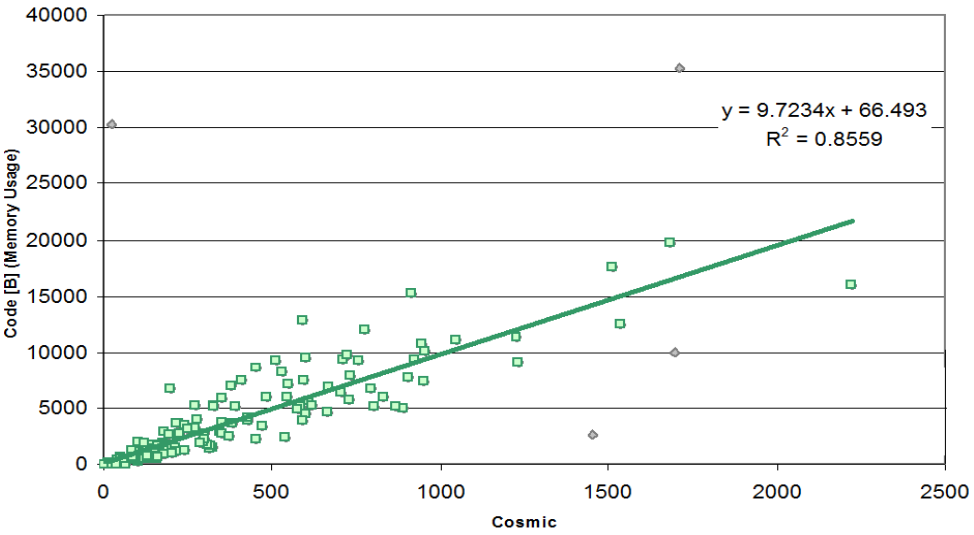


图 10. 代码规模（字节）与 COSMIC 功能规模(CFP)

参考文献

所有 COSMIC 文档和下面列表中标识有(*)的文档，包括翻译成其他语言，可以从 www.cosmic-sizing.org 下载目录下获得。

[1] Measurement Manual, v4.0, April 2014 and v4.0.1*.

[2] ISO 14143:2007 Software Engineering – Software Measurement – Functional Size Measurement, Part 5, Determination of Functional Domains for Use with Functional Size Measurement.

[3] Sommerville, I., Software Engineering, Addison-Wesley Publishing Company, 1995.

[4] EARS (Easy Approach to Requirements Syntax). 17th IEEE Engineers Requirements Conference, 2009.

[5] ISO 61131-3 Programmable controllers – Part 3: Programming languages, 2003, URL: www.iec.ch.

[6] Guideline for approximate COSMIC functional size measurement (*).

[7] Requirements Engineering Tools, IEEE Computer Society, 2010.

[8] Lind, K., Heldal, R., Harutyunyan, T., Heimdahl, T., CompSize: Automated Size Estimation of Embedded Software Components, IWSM conference 2011, Nara, Japan.

[9] Lind, K., Heldal, R., A Practical Approach to Size Estimation of Embedded Software Components, Approved for publication in IEEE Transactions on Software Engineering, 2011.

[10] Stern, S., Practical Experimentations with the COSMIC Method in the Automotive Embedded Software Field, section 4.2 of COSMIC Function Points: Theory and Advanced Practices, CRC Press, 2011, pp. 237-246.

[11] Soubra, H., Abran, A., Stern, S., Ramdan-Cherif, A., Design of a Functional Size Measurement Procedure for Real-Time Embedded Software Requirements Expressed using the Simulink Model, 21st International Workshop on Software Measurement – 6th International Conference on Software Process and Product Measurement – IWSM-Mensura 2011, Nara (Japan), Nov. 3-4, 2011, Editor: IEEE Computer Society, pp. 76-85.

[12] Soubra, H., PhD thesis Automation de la mesure fonctionnelle COSMIC-ISO 19761 des logiciels temps-réel embarqué, en se basant sur leurs 41pecifications fonctionnelles. Ecole de technologie supérieure (ETS), Université du Québec and Université de Versailles at St-Quentin, in collaboration with Renaults SAS.

[13] Lavazza, L., Del Bianco, V., A case study in COSMIC functional size measurement: the Rice Cooker revisited, IWSM/Mensura conference 2009 (*).

[14] Al-Sarayreh, K.T. and A. Abran, Specification and Measurement of System Configuration Non Functional Requirements, 20th International Workshop on Software Measurement (IWSM 2010), Stuttgart, Germany, 2010.

[15] Symons, C., Accounting for non-functional requirements in productivity measurement, benchmarking and estimating, UKSMA/COSMIC International Conference on Software Metrics & Estimating, London, UK, October 2011, URL: www.uksma.co.uk.

[16] Butcher, C., Delivering Mission-Critical Systems,,British Computer Society meeting, London, 18th November 2010.

[17] Toivonen, H., Defining measures for memory efficiency of the software in mobile terminals International Workshop on Software Metrics, 2002.

[18] COSMIC Rules for Embedded Software Requirements Expressed using Simulink, published by Renault 2012 (*).

[19] 'Automatic COSMIC sizing of requirements held in UML', Jaroslaw Swierczek, COSMIC Masterclass, IWSM 2014, Rotterdam (*)

[20] 'Guideline for 'Measurement Strategy Patterns' (*)

[21] Diab, H., Koukane F., Frappier M., St-Denis R. “µcROSE: Functional Size Measurement for Rational Rose RealTime.” (2002)
[22] Soubra, H., and Chaaban, K. “Functional Size Measurement of Electronic Control Units Software Designed Following the AUTOSAR Standard: A Measurement Guideline Based on the COSMIC ISO 19761 Standard.” (2012).
[23] Soubra, H. “Fast Functional Size Measurement with Synchronous Languages: An Approach Based on LUSTRE and on the Cosmic ISO 19761 Standard.” IWSM-MENSURA, 2013. (*)
[24] Guideline on how to account for Non-Functional Requirements in software project performance measurement, benchmarking and estimating’ (under development) *
[25] Soubra, H., Abran, A., Stern, S., Ramdan-Cherif, A., “Design of a Functional Size Measurement Procedure for Real-Time Embedded Software Requirements Expressed using the Simulink Model”. IWSM-MENSURA, Nara, Japan, 2011.
[26] Stern, S., Gencel, C., Embedded software memory size estimation using COSMIC: a case study, IWSM 2010 (*)

实时软件领域的术语

Actuator. 致动器：一种设备，将一个电信号转换为机械运动。

Clock. 时钟：（在本指南中使用的）一个装置，以恒定频率产生一连串的脉冲。

Control. 控制：指导和指引

Dumb device. 哑设备：根据接受的提示信息，自动发送数据的任何类型的设备。

注意：为了度量目的，作为软件需要的数据，一个输入数据移动可以从一个哑设备获取相关数据。

Intelligent device. 智能设备：必须被明确告知需要什么数据，从而产生出所需要的输出数据的任何类型的设备。

注意：为了度量目的，一个明确需要数据的软件的一个输出数据移动，被一个智能设备作为输入接收。设备发出一个输出运送需要的数据作为一个输入被所需要的软件接收。

Monitor. 监控器：保持对事物的检查。

Sensor. 传感器：将物理信号转换成一个电信号，提供软件可获取格式的数据组的设备。

Timer. 计时器：(如本指南中使用的)一个度量时间间隔的设备或软件过程。

附录 A – 从版本 V1.0 到 V1.1 的主要变化

这个附录包含这个“实时软件规模的指南”从版本 1.0 到现在 1.1 版的重大改变的总结。

V4.0 Ref 引用	从版本 1.0 的变化
前言	详细地讨论了在指南中被使用的“实时软件”的定义。
1.2.3	“电梯门关闭”案例 2 描述一个关闭序列需要两个功能处理，而不是一个，更切合实际。
1.2.4	引用 IEC 61131-3 标准已经被更新到 2013 版。
2.2	v1.0 的文字使用“时钟”和“定时器”时不一致（在案例 2.2，功能用户被作为“时钟（定时器）”）。这两个术语使用被合理化，并且他们现在被定义到术语表中。
2.2	案例 3c)错误地描述为有且仅一有一个功能用户。事实上，作为主管办公室的按钮有一个功能是生产线上按钮不能实现的，软件必须能将主管办公室的按钮与其他按钮区分出来，所以有两个功能用户。 同样的理由适用于案例 3d) 现在有 3 个功能用户。案例 3d)错误地引用到“如案例 b)”。现在已经正确地指向“如案例 c)”。
3.1	识别触发事件和功能处理的过程描述已经被修改以保持与 v4.0 和 v4.0.1 保持一致。
3.2.2	图 3.2 的左上角的注释 “从同样的功能用户来的同一个功能处理的触发输入和其他输入”。在实时软件环境下，可能会导致误解，因此已经被删除。在实时软件中，一个硬件或软件功能用户发起一个功能处理可能发送一个数据组传达描述一个触发输入的所有数据。它通常并不会发送任何后续数据描述不同兴趣对象。
3.2.4	增加了新的一节讨论在实时软件度量中的错误或故障消息。
4.1.2	PLC 的需求描述被重写，使其更加清晰。 在功能处理“停止泵”中两行已经变样。更安全的做法是在关闭出口阀之前停止泵。
4.2 (formerly section 3.2.4)	例 1 和例 2 已经纠正了对 “时钟”和“定时器”区分，使功能更加清晰。(参见上面的修正为 2.2 节)。 在图 3.3 中,“删除请求”被纠正为“写”。在文本中被错误描述地描述为“输出”。现在文本已经纠正状态为“写”。
4.3	在图 4.2 的环境图，“电源”被更容易理解的“电力”来替。
4.3	使用 “定时器”代替“时钟”。
4,5	在轮胎压力监测系统的环境图中“持久存储器”被移除。（它没有被使用，同时解释也改进了）。
4.5	在环境图 4.5 中，“定时器”代替了“时钟”；同时，“点火钥匙”已被删除，因为它不是这个场景中的功能用户。转动点火钥匙启动时钟控制 TPMS 软件。
4.7	添加了一个新章节，提供了实时软件的一个案例，它显示了软件移动的计数是一个

	很好的软件功能的度量元，因为它可以合理地代表数据运算的功能。
4.8	一个新章节提供了如何成功将 COSMIC 方法用于估算车辆电子控制单元的内存规模需求的例子。
术语表	术语中增加“时钟”和“定时器”的定义。

附录 B

附录 B - COSMIC 变更请求和建议程序

COSMIC 度量实践委员会（MPC）非常愿意接受反馈、建议，以及对此指南的变更请求（如果必要的话）。本附录阐述了如何与 COSMIC MPC 联系。

所有与 COSMIC MPC 的联系都可以发送电子邮件到下面的地址：

mpc-chair@cosmic-sizing.org

非正式的反馈和建议

关于此指南的非正式建议和/或反馈，比如理解或应用 COSMIC 方法的任何困难，一般性改进的建议等，都可以通过电子邮件发送到上面的地址。消息会被登记下来，并且通常在收到后的两周内给予答复。度量实践委员会不保证对一般性意见给予答复。

正式的变更请求

本指南的读者如发现某个文字缺陷，或不足之处需要澄清，或者一些文字需要加强，那么可以提交一个正式的变更请求（“CR”）。正式的 CR 会被登记下来，并在收到后的两周内给予答复。每个 CR 将分配一个序列号，在 COSMIC MPC 的成员中循环传递，COSMIC MPC 是一个世界范围内 COSMIC 方法的专家组。他们的正常评审周期最少需要一个月，如果变更请求很难解决，可能需要更长的时间。评审的结果可能是 CR 被接受，或者拒绝，或者“未决待进一步讨论”（后面这种情况，例如本 CR 要依赖另一个 CR），结果会尽快地反馈给提交者。

只有包含了下面的所有信息，一个正式的 CR 才会被接受。

- 提交CR的人员姓名、职位和单位
- 提交人的详细联系方式
- 提交日期
- 关于CR的目的的总体陈述（如“需要改进文本……”）
- 需要变更、替换或删除的实际文字（或澄清引用出处）
- 建议增加或替换的文字
- 关于变更的必要性的充分解释

提交 CR 的表格可以从门户网站 www.cosmic-sizing.org 获得。

COSMIC MPC 对 CR 的评审结果，以及在哪个版本应用这个 CR（如果被接受了的话）的结论是最终的决定。

对 **COSMIC** COSMIC 方法应用的问题

COSMIC MPC 抱歉不能回答与 COSMIC 方法应用相关的问题。有商业机构能够提供本方法的培训和顾问工作，或者支持工具。详细情况请查询 www.cosmic-sizing.org 网站。