COSMIC 度量手册

ISO 19761

第三部分:

案例

5.0 版本

2020.5

前言

COSMIC 度量手册（ISO 19761:2011）包括三个部分：

第一部分：原则、定义&规则

第二部分：指南

第三部分：COSMIC 概念和度量活动的案例

本文档为第三部分，包括了 COSMIC 功能规模度量方法（简称 COSMIC 方法）的概念和度量案例。案例按照每个度量阶段划分，案例的背景和目的用斜体表示。

COSMIC 度量手册和技术报告，包括其它语言的翻译版，可以在门户网站 www.cosmic-sizing.org 上找到。

编辑：

Alain Abran，魁北克大学（加拿大）
Peter Fagg（英国）
Arlan Lestherhuis（荷兰）

COSMIC 度量实践委员会的其他成员：

Jean-Marc Desharnais，魁北克大学（加拿大）
Cigdem Gencel，博尔扎诺自由大学（意大利）
Dylan Ren，麦哲思科技（北京）有限公司（中国）
Bruce Reynolds，Telecote 研究所（美国）
Hassan Soubra，开罗德国大学（埃及）
Sylvie Trudel，魁北克蒙特利尔大学（加拿大）
Frank Vogelezang， Metri 集团（荷兰）

中文版翻译者&校对者：

郭玲，麦哲思科技（北京）有限公司

任甲林，麦哲思科技（北京）有限公司

夏思文，易才人力资源公司

张坤，上海千杉网络技术发展有限公司

高艳，无锡农村商业银行

程敏，中汇信息技术（上海）有限公司

徐妍玲，麦哲思科技（北京）有限公司

目录

1 COSMIC 介绍

1.1 功能性用户需求（FUR）

关于功能性用户需求（FUR）的大量案例，请免费查阅 cosmic-sizing.org 中的案例研究。

1.2 非功能性需求（NFR）

COSMIC 方法只适用于 FUR 的规模度量。并不适用于需求中的非功能性需求（NFR）。但是最开始作为 NFR 的需求，随着项目的进展，很可能会演变为 FUR。因此，区分这两类需求以及它们的演变很重要。

业务应用软件案例：新软件系统的需求包括“用户需要通过加密的方式来保证文件安全”。目前该项目处于估算项目工作量和成本的阶段。考虑了以下两种方案：

- 开发专门加密的软件。为了进行项目估算，可能需要度量加密软件的 FUR 的规模。
- 购买现成的商品现货（COTS）程序包。为了进行项目估算，可能只需要度量集成 COTS 程序包所需的软件功能的规模。在估算项目成本时，也应该考虑到程序包的成本以及集成、测试文件加密程序包的工作量。

实时软件案例：航天系统的可靠性或容错性是通过系统的冗余设计及物理备份来实现的。比如，一个引擎监控的功能，被实现在多台独立的嵌入式计算机里。这个功能作为一个有严格时间约束的*NFR*，被描述为：“每一台独立的计算机都必须在特定的时间里做出响应。如果其中任何一台计算机总是晚于规定时间做出响应，或其显示的结果与其他计算机不一致（描述转化为了功能需求），该计算机必须被淘汰。” 因此，一个最初呈现为关于容错的非功能需求演变成了可度量的功能性用户需求。这种时间控制机制也可以通过软件实现一部分，而且这一功能也可被度量（请查阅 “度量实时软件规模的指南”3.2节中的案例）。

1.3 COSMIC 软件环境模型

见 cosmic-sizing.org 中的案例研究。

1.4 通用软件模型

见 cosmic-sizing.org 中的案例研究。

1.5 类型与实例

COSMIC 方法只涉及到事物的类型，而不涉及到该类事物发生的次数（实例）。这条原则对规模的影响很大，适用于度量活动所涉及到的所有事物，如功能用户、兴趣对象、数据组等。

软件环境模型的案例。

业务应用软件案例 1：一个支持 100 名员工的呼叫中心系统，负责接听处理客户问题。针对所有的员工，需求都是一样的（回答客户的问题），此系统的环境模型只有一个功能用户类型：“呼叫中心的员工”，此类型有 100 个实例。

实时软件案例 1：一个数字无线电的嵌入式软件会将其输出发送给一对立体声扬声器。该软件向这两个扬声器分别发送相同类型的信号，它们都用同样的方式把接收的电信号转化为声音。此软件的环境模型只有一个名为“扬声器”的功能用户类型，此类型有 2 个实例。

通用软件模型的案例。

业务应用软件案例 2：假设一个功能处理用于新用户数据的输入和验证。当一个人类功能用户注册新用户数据时，这个功能处理会被执行，即它会发生一次。在它执行过程中，通过搜索数据库检查客户是否已存在，确认输入数据的读数据移动发生一次或多次（这取决于数据库设计）。但是，在度量该功能处理时，只计数一个输入和一个读。

实时软件案例 2：某个功能处理要求每隔 10 秒监控烤箱温度，即该功能处理每 10 秒钟被执行一次。在执行期间，根据烤箱温度判断，是否需要打开/关闭加热器，或保持状态不变。在该功能处理中，无论是否打开/关闭加热器，对输出的数据移动计数为一次。

1.6 COSMIC 度量过程

请参考 cosmic-sizing.org 中的案例研究。

2 度量策略阶段

2.1 度量目的

度量活动是为了满足干系人的信息需要。为了满足干系人的需要，度量人员必须要了解度量目的。

举例：下面这些是典型的度量目的。

- 当 FUR 逐步演变时，度量其规模作为估算开发工作量的输入。
- 度量 FUR 的变更规模，管理额外的需求变更导致的“范围蔓延”。
- 度量已交付的软件规模，作为组织性能的度量输入。
- 度量已交付的整体软件规模和已开发的软件规模，了解功能复用的情况。
- 度量已有软件的规模，作为度量软件维护团队性能的输入。
- 度量已有软件的某些变更的规模，作为变更项目团队输出的成果物规模。
- 度量软件功能中必须被开发提供给人类用户使用的功能规模。

2.2 度量范围

度量范围可以是整个软件或部分软件，这取决于度量目的。

业务应用软件案例：图 2.1 展示了某项目团队交付的所有软件块，即“整体范围”：

- 一个已实现的应用程序包的客户端和服务器端构件。
- 在现有应用与新程序的服务器构件之间的接口程序。
- 数据转换的程序，用于按照需要将现有数据转换为新格式。此程序由一些项目组开发的可复用对象构建而成。
- 新硬件设备的驱动软件，包括了客户端构件。

对于每个独立软件块的度量范围，使用一个矩形方框来表示。

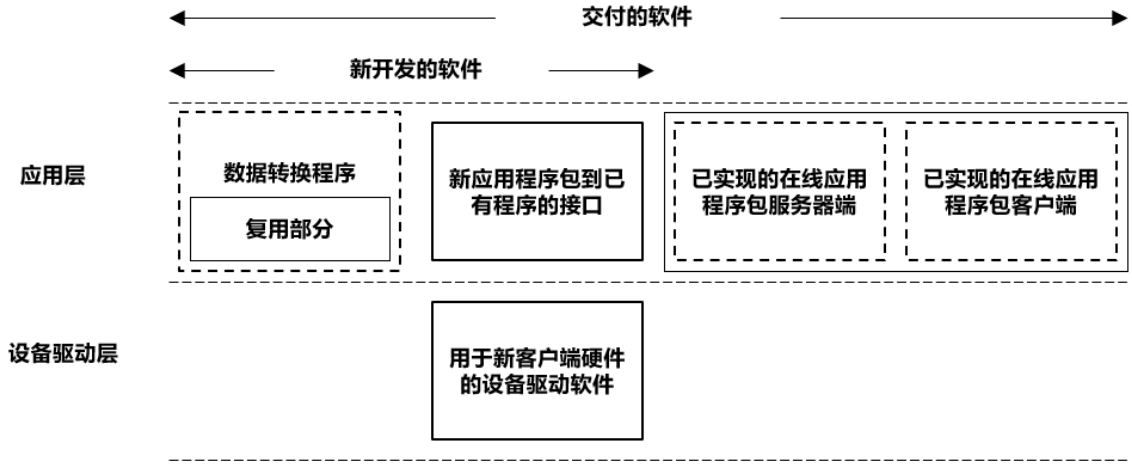

图 2.1 – 软件交付的总体范围和单个度量范围

通过上图可知，该软件中的部分组件是新开发的，部分组件是之前开发的（已实现）。度量目的是把软件包作为一个整体来度量软件的 FUR，忽略客户端-服务器端的结构。

已实现的程序包规模加上接口程序的规模，是更新后的总规模。由于数据转换程序只被使用一次就废弃，其规模无需度量。但是在度量规模时，应该考虑复用组件的规模，还有新开发的设备驱动的规模。

由于可交付成果的多样性，在衡量整个项目团队的性能时，将所有交付软件的大小相加是不明智的。交付每个软件的团队的性能应该单独测量。

需要注意的是，通常我们只需要度量软件中需要实现的那部分，而不是软件本身。对后者的度量，可能软件供应商更感兴趣。

2.3 功能用户

对于同一功能处理来说，某功能用户可能会出现多次，以进行数据输入。但是，COSMIC 方法只考虑类型而不考虑实例（见 1.5）。

业务应用软件案例 1：一个订单系统有很多员工（人类功能用户）负责维护订单数据。在他们输入的所有数据组中都增加员工 *ID* 一项。本例应识别为一个员工功能用户类型，因为订单系统 *FUR* 对于所有员工是相同的。

实时软件案例 1：汽车的每个轮子都有一个传感器用于监测其轮胎压力。每隔一段时间，一个功能处理必须获得四个轮胎的压力值。如果压力过小或过大，即超出安全压力范围，仪表盘屏幕的一个有四个轮胎的图标会显示哪一个轮胎出了问题。这里的功能用户是四个传感器和显示屏上的图标，由于四个传感器属于相同类型，图标也属于相同类型，因此只要识别一个“传感器”功能用户类型以及一个“图标”功能用户类型。

不同的功能用户需要不同的功能。

业务应用软件案例 2：一个软件系统的功能是维护基本的人力信息，人力资源部门的所有员工均可查阅该信息，而对于敏感的薪资信息只有其中部分员工可查阅。因此，该软件有两个功能用户类型。该软件的度量目的应该定义度量范围是所有人员都可以访问的全部功能，还是仅限于其中一类员工可以访问的功能。

FUR 是从功能用户的视角来编写的。因为每个功能用户可能从不同的角度解读系统，因此需求也会不一样。重要的是，所有待度量的需求都从同一个功能用户视角来描述，以便于规模之间的比较。

实时软件案例 2：一个复印机嵌入式软件的功能用户可以通过两种方式确定。可以定义为（a）想要进行复印的人类用户，或者（b）复印机的硬件设备，如控制按钮，向人类用户显示信息的屏幕，送纸系统，卡纸传感器，油墨控制器，指示灯等与软件直接交互的硬件。这两种功能用户类型，人类或者硬件设备所“看到”的功能是不同的。比如人类用户，只能了解到复印机的一小部分功能。开发复印机驱动程序的嵌入式软件开发人员需要把硬件设备作为其功能用户。而市场人员为了比较产品的价格/性能[1]，可能会认为度量人类用户能看到的那部分复印机功能更有用，以便与竞争对手的复印机功能进行比较。但是，不要将这两种视角混为一谈，既从人类功能用户角度又从硬件设备角度来度量规模是非常困难的。

2.4 度量策略模式

更多例子请参考度量策略模式指南。

2.5 层

软件的通常架构是其功能分布在多个组件上。组件间按照其需求规定的规则传递消息以进行交互。反过来，处理相似问题的组件也可能包含在一个层中。在一个层中，组件可以是对等软件块、客户端或服务器端。根据 COSMIC 方法，只有处在同一层的组件规模可以互相比较和累加。

案例 1：图 2.2 中处于应用层的软件块都是相互对等的。

案例 2：一般在软件体系结构中，“顶”层，即在整个分层结构中不从属于任何其他层的层，通常被称作“应用程序”层。这个应用层里的软件依赖所有其他层的软件提供的服务来正常运行。处于此“顶”层的软件可以再进行分层，例如由用户界面、业务规则以及数据服务构件所组成的“三层结构”（参见下文的例 2）。

业务软件案例 *1*：图 2.2 展示了业务应用软件典型的分层软件体系结构的物理结构：

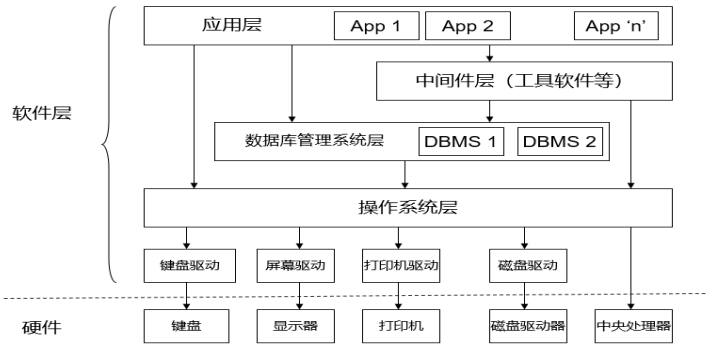

图 2.2 – 商业/MIS 计算机系统的典型分层软件体系结构

[1] 例如，2002 年 10 月，在德国马格德堡举行的国际软件测量研讨会上，Toivonen 在“移动终端软件内存效率的度量定义”中比较了仅供人类用户使用的手机功能的大小。

实时软件案例 1：图 2.3 展示了嵌入式实时软件典型的分层软件体系结构的物理结构。（注：简单的单任务实时嵌入式软件可能不需要实时操作系统。）

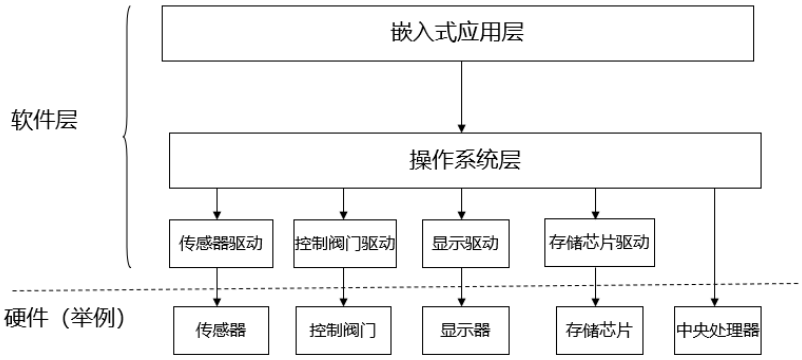

图 2.3 – 典型的嵌入式软件的层次架构

实时软件案例 2：通信系统的 ISO 7 层（OSI）模型定义了一种分层体系结构：信息接收层软件的层次通信规则与信息发送层的规则是相反的。

实时软件案例 3：在汽车工业中，“汽车开放系统架构”（AUTOSAR）展现了层与层之间所有不同类型的通信规则，也是使用层的原理描述的。

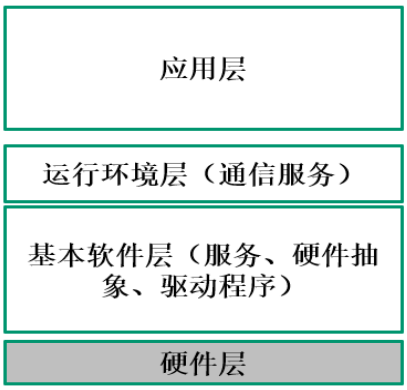

图 2.4 – AUTOSAR 架构

软件架构展示的层，根据“视角”不同而不同。

业务应用软件案例 2：某应用软件位于分层的架构中，如下图 2.5，根据不同的“视角”可以有不同的层结构。

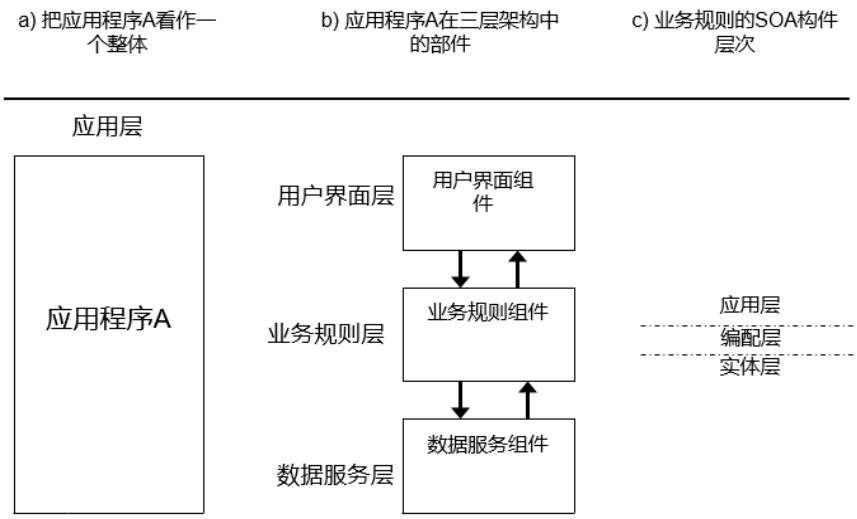

图 2.5 -应用程序不同的层结构划分

目的 1：把应用软件 A 作为一个整体来度量规模，如视图 a)。度量范围是整个软件，完全存在于应用层。

目的 2：应用软件 A 是基于三层架构来搭建的，用户界面、业务规则和数据服务组件。目的 2 是分别度量这三个组件，如视图 b)。每个组件都位于其三层架构中各自的层中，因此度量范围必须分开定义。

目的 3：应用程序的业务规则构件使用了面向服务架构 SOA）的可复用构件进行构建，它有自己的层次结构。如视图 c)，目的 3 是要度量业务规则构件的 SOA 构件。每个 SOA 构件都处于 SOA 结构中的一层内，各构件的度量范围必须分开定义。（注意：SOA 术语在自己的体系结构中也使用“应用层”一词。）

2.6 分解层级

软件可能包括多个层级，在 COSMIC 中称为分解层级。因为软件块组件的规模只可以与处在相同分解层级的组件直接比较，因此根据度量目的，在恰当的分解层级上进行度量十分重要。

案例：“度量策略模式”指南中归纳了三个标准的分解层级：“应用整体”、“主要构件”和“次要构件”。请查阅上面的业务软件案例 2，该例中的图 2.5 展示了这三种级别。

2.7 环境图

环境图有助于可视化展现在功能用户和持久存储介质环境中的待度量软件。

业务应用软件案例：图 2.6 展示了客户端/服务器端的软件，其实现的应用程序包如图 2.1 中的例子所示，把待度量的软件作为一个整体，即忽略应用程序包是由两部分组成（客户端和服务器端）。

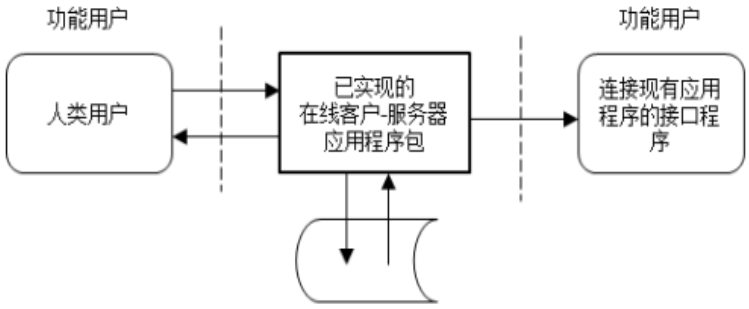

图 2.6 – 客户-服务器应用程序的环境图

实时软件案例：图 2.7 显示了一个简单的嵌入式防盗报警软件系统的环境图。

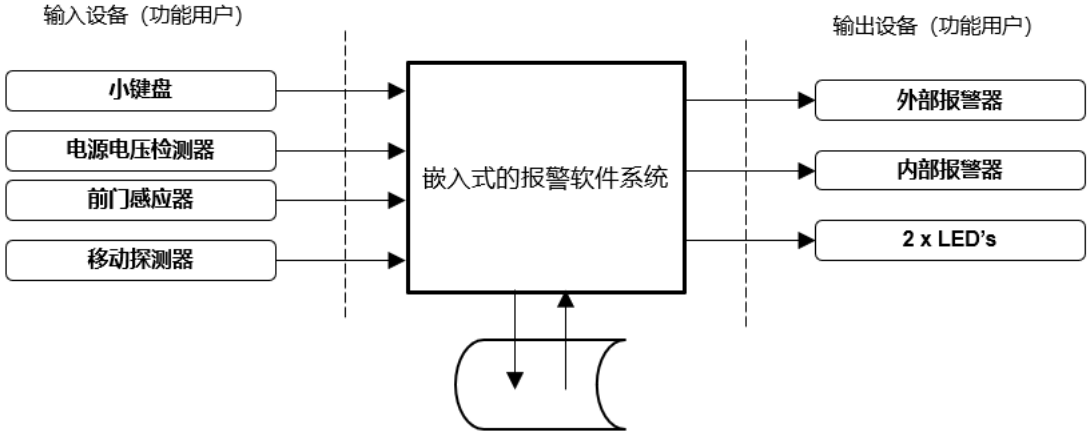

图 2.7 – 嵌入式防盗报警软件系统的环境图

2.8 颗粒度等级

在开发阶段的早期，需求随着信息的增加和理解的深入而逐渐完善。在早期阶段，功能处理以及识别和度量它们的所需信息可能都不存在。在这种情况下，可以使用多种方法来估算规模，见“COSMIC 早期软件规模度量指南”。功能性用户需求可能处于较高的颗粒度级别，可能描述了功能用户组，而不是独立的人类用户或工程设备或软件块。

在准备使用 COSMIC 方法度量需求前，要确保需求必须处于功能处理颗粒度级别，这是非常重要的。

案例：一组功能用户可以是其成员处理多种类型功能处理的“部门”，也可以是具有多种类型仪表的“控制面板”，或叫“中央系统”。

一个事件组可能在比较高的颗粒度上描述了一组功能需求，比如会计软件系统的输入流标记为“销售业务”，或是航空软件系统的输入流标记为“飞行员指令”。

实时软件案例：关于在不同颗粒度级别以及不同分解层级的规模度量的案例，请见早期软件度量指南中的电信系统的案例。

业务软件案例：这个实例来自于业务应用软件领域，是一个知名的网上购物系统的一部分，我们称之“珠峰订货系统”。本案例的目的是展示不同的颗粒度级别，从而发现不同级别下的功能处理。为了理解不同的颗粒度级别，以下描述均为高度简化。

如果我们想要度量这个系统，我们可以假设度量目的是要确定该系统中供人类功能用户使用的这部分的规模。然后我们将度量范围限定在“客户可以通过互联网订购珠峰系统中的商品”的这部分。请注意，本例的目的是要阐述不同级别的颗粒度级别，因此，我们将只探讨系统总体功能的某些部分，这些部分足以理解粒度级别的概念。本案例是关于 FUR 颗粒度级别，而不是分解层级。

在该系统最高的第 1 级（主要功能）需求，只包括简单的概括性描述，如下。

“珠峰订单系统必须能够使客户可以查询、选择、订购、支付和配送珠峰产品范围内的所有产品，包括第三方供应商提供的产品。”

把这个最高颗粒度级别的需求进一步细化，我们发现第 2 级需求包含了四个子功能，如图 2.8（a）所示。

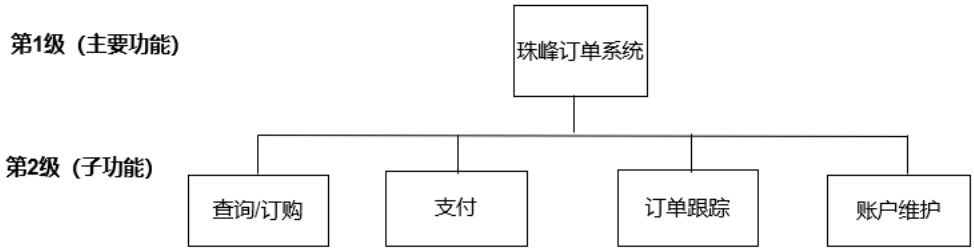

图 2.8（a） - 珠峰订单系统分析：第 1 级&第 2 级颗粒度

四个子功能需求如下：

- 查询/订购子功能，允许客户查询珠峰数据库里的所有产品的价格和是否可购买，并添加到购物车。该子功能还可以展示打折商品、提供所选商品的评价和进行一般查询，如查看配送说明等。这是一个非常复杂的子功能。因此在第 2 级颗粒度级别下，我们不再进行进一步的分析。
- 支付子功能允许客户提交订单并支付购物车中的商品。
- 订单跟踪子功能允许客户查询当前订单的进展及配送情况，并可对订单进行修改（如修改配送地址），或退回不满意的商品。
- 账户维护子功能允许当前客户维护其账户信息，如家庭地址、支付方式等。

图 2.8（b）和（c）进一步展示了支付子功能、订单跟踪子功能和账户维护子功能的需求。在这个需求逐渐细化的过程中要注意：

- 我们并没有改变待度量功能的范围，
- 所有颗粒度级别的功能描述都是客户可用的功能，即客户可以在所有这些粒度级别上“看到”应用程序的功能。

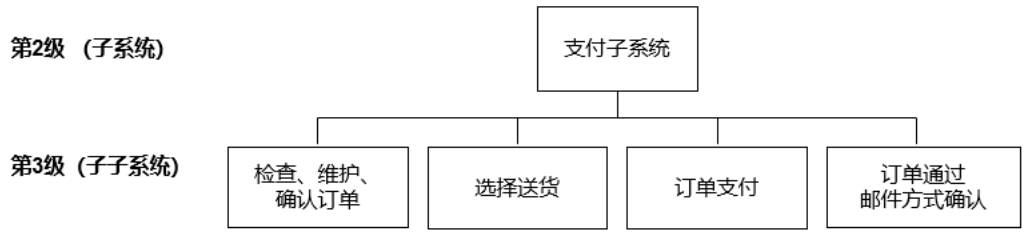

图 2.8 (b) - 关于支付子功能的分析

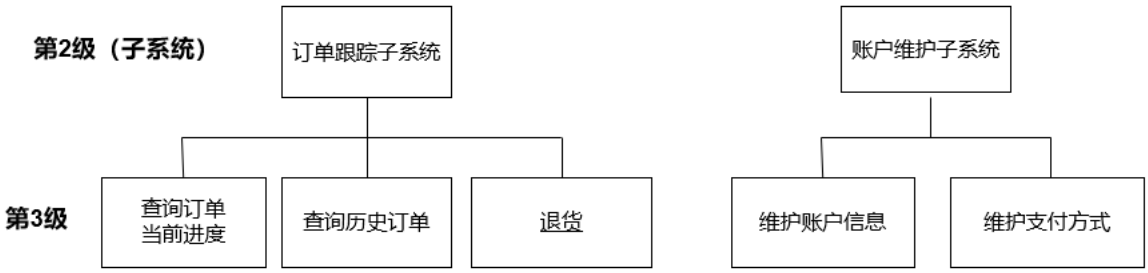

图 2.8 (c) - 关于订单跟踪和账户维护子功能的分析

图 2.8（c）展示了我们进一步细化订单跟踪子功能需求到第 3 级颗粒度级别，我们发现了两个功能处理类型（两个查询属于同一类型，以及退货）。如果我们对需求进一步细化到更低的颗粒度级别，可能会发现更多的功能处理。因此，本例说明了当对需求进行“自顶向下”的细化时，某个特定级别上的需求并不一定与 COSMIC 方法中定义的颗粒度级别一致（COSMIC 方法中定义为功能处在一个“相对较细的颗粒度级别”）。

此外，其他分析人员可能会用不同的方式绘制图表，在图表中的每个颗粒度级别中显示其他类型的功能。对于如此复杂的系统，并没有一个绝对正确的划分方法。

鉴于在实践中这些变数是不可避免的，度量者必须检查和分析图标中的各个颗粒度级别，以确定待度量的功能处理。如果在实践中无法做到，如需求无法详细到功能处理的颗粒度级别，则必须使用近似度量方法。我们通过账户维护子功能分支中的“维护客户信息子子功能”来说明，见图 2.8（c）。

对于经验丰富的度量者来说，“维护”这个词几乎毫无例外的代表一组事件也就是一组功能处理。因此，我们可以假设“维护”子功能必定包含三个功能处理，称为“查询顾客信息”、“更新顾客信息”和“删除顾客信息”。（当然必须有“创建客户详细信息”的功能处理，但这属于系统的另一个分支，当客户首次购买商品时需建立账户。这在本例的范围之外。）

经验丰富的度量者应该能够用 COSMIC 功能点为单位来“推测”这个子子功能的规模，先假设功能处理的数量（本例为三个），然后用这个数乘以功能处理的平均规模。平均规模可以通过对这个系统的其他部分或者其他相似系统进行基准度量而得到。基准度量过程的实例在“COSMIC 早期软件规模度量的指南” 中给出，此文档还包含了近似规模度量的其他方法的实例。

当然，这种估算方法也有局限性。如果我们对第 1 级需求（“珠峰订单系统必须允许客户查询、选择、订购、支付和配送珠峰产品范围内的所有产品，包括第三方供应商提供的产

品。”）使用这种方法进行估算，我们只能识别出少量的功能处理。但是进一步分析可能会发现，在这个复杂的系统里实际上存在很多功能处理。这也就是为什么功能规模会随着需求的细化而逐渐增加，即使并没有改变范围。因此，对于高颗粒度级别的需求，即很模糊的需求，必须小心使用这些估算方法。因此，当可获得的细节非常少时，在高颗粒度级别上使用这些近似方法必须非常小心。

3 映射阶段

3.1 功能处理

在 COSMIC 方法中，如果要获得一个正确的规模度量结果，功能处理的识别是必不可少的。下面是如何识别功能处理的案例。

人类功能用户输入数据，将启动一个功能处理。

业务应用软件案例 1：某公司收到一个订单（触发事件），由一位员工（功能用户）负责输入订单信息（触发输入，包含了关于兴趣对象“订单”的信息），这是“订单输入”功能处理的首个数据移动。

业务应用软件案例 2：人事系统软件中的某个功能处理的触发输入中的数据组描述了一个新员工。该数据组是由操作人事软件的人类功能用户输入的数据。

设备功能用户传递数据，将启动一个功能处理。

实时软件案例 1：实时软件系统中的某个功能处理是由其触发输入启动的，该触发输入仅仅告知功能处理产生了一个时钟节拍（功能用户）。移动的数据组所传递的数据（节拍，可能只用一个比特表示）只是告知发生了某个事件发生。

实时软件案例 2：工业实时火灾探测系统中的某个功能处理的触发输入，可能由特殊的烟雾探测器（功能用户）触发。检测器生成的数据组传递了信息“发现烟雾”（事件发生），并包括了检测器的 ID（可以用于确定事件发生位置的信息）。

实时软件案例 3：在超市的收银台，当一个商品的条形码靠近条形码阅读器（功能用户）的屏幕时（触发事件），阅读器会生成一个数据组，包括了条形码的图片，这是作为收银台软件的输入。数据组图片由触发输入移动给其功能处理。如果条形码有效，后者将把产品的价格添加到客户账单上，并发出“哔”一声，告知客户产品已扫描并记录了销售信息。

识别独立的触发事件，进而识别出对应独立的功能处理。——当一个人类功能用户在软件之外做出“下一步该做什么”的决定时，这个决定在时间上是独立的，并且需要得到软件的独立响应。

业务应用软件案例 3：一个功能用户输入了客户订单信息，订购复杂的工业设备，接着向客户确认订单已接受。在输入订单信息和确认接受订单之间，功能用户可能会进行一些查询操作，比如订单是否可以在预定的时间内交付、用户的信誉情况等。尽管在输入订单信息之后必须向客户反馈订单已接受，但是在这种情况下，功能用户需要做出独立的决策，是否接受订单。因此，这表明订单输入和订单接受是两个独立的功能处理。

当活动职责独立时，识别独立的触发输入和功能处理。

业务应用软件案例 4：在人事系统中，负责维护基本人事信息的职责通常与负责维护工资信息的职责是分开的，即功能用户不同，也需要分开识别功能处理。

业务应用软件案例 5：假设在业务应用案例 1 中收到的订单，订单处理系统需要向客户注册系统发送新客户的详细信息，而我们度量的范围是客户注册系统。那么此时，订单处理系统

将作为客户注册系统的功能用户。订单处理系统接收到新客户的信息后，生成客户信息数据组，并发送给客户注册系统，这个动作触发了一个存储这些信息的功能处理。

当度量功能处理的规模时，不需要区分功能处理是否是在线处理或批处理。

业务应用软件案例 6：假设业务应用软件案例 1 中的订单是通过“线下”的方式输入的，比如通过对纸质文档进行扫描，并且是临时存档以进行自动的批处理。如果订单输入功能处理是批处理的方式执行，应该如何分析？功能用户是在线下输入订单数据并准备好进行批处理的人；进行订单批处理的触发输入是功能处理的一个数据移动，此数据移动把订单数据组移动到处理中。（如果一定要度量线下处理，它涉及另一个独立的功能处理：把订单载入到临时存储器内）。

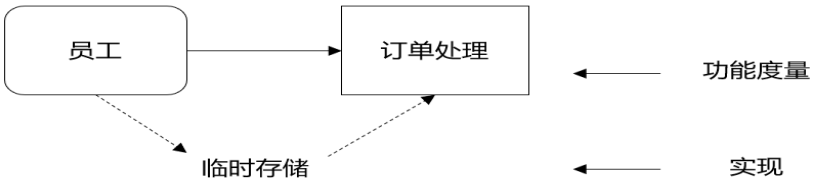

图 3.1 – 业务应用案例 4 批处理

业务应用软件案例 7：假设一个年终批处理应用程序的 FUR 是汇报本年的经营成果，并重置结果以供下一年使用。理论上，操作系统生成的年终时钟节拍启动了这个应用程序。但逻辑上，该应用程序中的每个功能处理都要从待批处理的数据流中获取输入数据。这应该以常规的方式来分析（如：一个功能处理包含一个或多个输入，第一个输入作为该功能处理的触发输入）。

然而，假设该应用程序中有一个特殊的功能处理，并不需要任何输入数据便能输出报告。理论上，人类功能用户授权操作系统触发该功能处理。因为每个功能处理必须有一个触发输入，我们可以把启动了批处理流的年终时钟节拍作为触发输入。接下来，该功能处理可能需要多次读和写，来生成报告。逻辑上，不管人类用户是通过鼠标点击一个在线的菜单栏来生成报告，或是授权操作系统来触发批处理来生成报告，都没有什么区别。

在实时应用程序领域中，传感器通常触发了一个功能处理。

实时软件案例 5：当一个传感器（功能用户）检测到温度达到某个值时（触发事件），传感器会发送一个信号，启动某功能处理的触发输入数据移动，关闭加热器（另一个功能用户）。

实时软件案例 6：军用飞机有一个探测“导弹接近”事件的传感器。传感器是软件的功能用户，必须对威胁做出响应。对于该软件，当且仅当传感器检测到威胁时，生成一个数据组，启动触发事件，比如“2 号传感器检测到导弹”，再加上导弹接近速度及其坐标的数据流。

基于数据输入的组织识别功能处理，或者检查部分已安装软件的菜单。

业务应用软件案例 8：假设某功能性用户需求有两类社会福利，一类是对多生一个孩子的家庭福利，第二类是对于低收入者的税收减免。这两类软件需要响应的事件是各自独立的。因此需要识别两个功能处理，尽管可能是通过一张表格来填写这两类信息。

只识别功能处理的数据移动：可能会有多种处理的路径，但并不一定是独立的功能处理。

业务应用软件案例 9：为数据库提供常规搜索功能的功能处理可能需要接受最多四个搜索参数（触发输入的属性）。但是如果仅输入一个/两个/三个搜索属性的值，都执行的是同一个功能处理。

业务应用软件案例 10：对于为汽车租赁公司注册新用户的功能处理来说，大多数数据属性是必须输入的，有部分属性是可选的。无论是输入全部属性还是一部分属性，对于新用户注册只有一个功能处理。

业务应用软件案例 11：接着案例 10，该公司另一个功能处理是进行汽车租赁预定，有多个选项可选，比如是否额外保险，是否有其他驾驶员，是否需要儿童座椅等等。选择不同的答案会流转到该租赁预定功能处理中的不同流程，但是对于汽车租赁预定来说仍然只有一个功能处理。

实时软件案例 12：航空系统中某功能处理的触发输入（由定位系统发送的飞机高度信息），根据其高度值（是否高于/低于某值）将有两个不同的处理路径。不同的路径会在飞行员地图上显示不同的数据组，如果高度太低，会额外发出警告。这里也只是一个功能处理。

3.2 数据组和兴趣对象

数据组和兴趣对象以各种形式存在于需求之中，从被度量的功能处理的需求环境中识别出它们，是度量人员的责任。

举例 1：实际上，数据组可能有很多来源，比如：

a) 一个在硬件存储设备上的数据结构（文件、数据库表、ROM 存储器等）。

b) 一个计算机易失性存储器中的一个数据结构（动态分配的数据结构，或者是内存空间中预先分配的一个内存块）。

c) 一个与功能有关的数据属性在 I/O 设备上(显示屏幕，打印的报告，控制面板显示器)的集中展现。

d) 一个在设备与计算机之间、或在网络中传输的一条消息。

举例 2：如果 FUR 中要求从只读存储器中读取数据，那么只读存储器就是持久性存储介质。

业务应用软件案例 1：在业务应用软件中，兴趣对象可以是“员工”（物理上），或者“订单”（概念上）。对于“订单”对象，通常从 FUR 的多行订单中可以识别两个兴趣对象：“订单”和“订单明细”。相应的数据组可以是“订单数据”和“订单明细数据”。

业务应用软件案例 2：假设 FUR 中针对数据库有一个特殊的功能处理，查询大于某年龄值的员工个数，某年龄值作为输入。该输入参数（年龄值）是一个数据组，定义的兴趣对象是“大于某年龄的员工集合”。输出数据组，包括了大于某年龄值的员工个数，描述的是与输入相同的兴趣对象。

业务应用软件案例 3：假设与案例 2 相同的查询功能处理，但是在输出时，除了要输出大于某年龄值的员工人数，还需要列出这些员工的姓名。相比于案例 2，现在要输出两个数据组：员工个数和员工姓名（来自持久性存储介质）。这是两个独立的数据组，因为他们的发生频率是不同的（个数统计值的发生频率为 1； 员工姓名的发生频率为 0，1，或多个）。

业务软件案例 *4*：*FUR* 中提到的兴趣对象的公共数据结构，是由功能处理来维护，并且能被大部分的功能处理访问。

识别实时软件领域的数据组和兴趣对象。

实时软件案例 1：从物理设备输入软件的数据组告知了设备的当前状态。在这种情况下，设备是兴趣对象（也是功能用户），而传递其状态的数据组（比如阀门是开还是关），启动了一个功能处理。类似的，输出到设备的数据组，如警告灯的开或关，传递了灯这个兴趣对象的状态信息。

实时软件案例 *2*：一个报文交换软件系统，根据其具体的软件 *FUR*，可能会将收到的一个报文数据作为一个输入，然后不作修改地作为一个输出推送出去。报文数据组的属性可以如：*“*报文 *ID*、发送者 *ID*、接收者 *ID*、路由码和报文内容*”*，兴趣对象是 *“*报文*”*。

实时软件案例 3：一种指针型数据结构，可能代表了多个兴趣对象，在 FUR 的表格中给出了属性值，被存放在持久存储器（例如 ROM）中，并且能被待度量软件中的大多数功能处理访问的。

实时软件案例 4：文件，通常指“平面文件”，可能代表了 FUR 中的多个兴趣对象，被存放在存储设备上。

功能用户也可能是兴趣对象。

业务应用软件案例 5：人类作为功能用户在登录功能处理中向系统输入了 *ID* 和密码。输入数据组对应的兴趣对象就是人类用户。

实时软件案例 5：假设温度传感器 *A* 发送某物质的当前温度给某功能处理。传感器提供关于其自身状态的信息，因此它也是这些信息数据组的兴趣对象。

3.3 数据属性

COSMIC 方法没有要求必须识别与数据组相关联的数据属性。但是，在指南的规则 13 和 14 中——第 2 部分的 3.4 节数据移动的唯一性中需要识别数据属性。

业务应用软件案例：“员工”兴趣对象对应的数据组可能命名为“员工主数据”，该数据组中包含的属性有“员工 ID”，“姓名”，“地址”，“出生日期”，“性别”，“婚姻情况”， “社保号码”，“级别”，“职位”等。

实时软件案例：温度传感器可以响应报告其属性“温度”的请求。安全传感器可以检测入侵者，并发送属性“检测到入侵”，传输的消息可以由“来源（地址），去向（地址），内容”数据属性组成。

3.4 数据移动

数据移动和时钟节拍。

实时软件案例 1：将每 3 秒发生一次的时钟节拍事件，识别为移动了仅包含一个属性的数据组的输入。这里的兴趣对象（和功能用户）是时钟，数据组传递了时钟的状态。

数据移动以及从系统时钟获得日期或时间。

业务应用软件案例 1：当一个功能处理将时间戳添加到一个要持久化或要输出的记录时，不会被识别为输入或输出。按照惯例，获得系统时间是操作系统提供给所有功能处理的功能。

数据移动和与功能用户的兴趣对象不相关的数据。

业务应用软件案例 2：不要将显示应用程序的通用数据识别为数据移动，如所有屏幕上出现的页眉和页脚（公司名称、应用程序名称、系统日期等）。

业务应用软件案例 3：不要识别移动控制命令（仅在业务应用程序领域中定义的概念）的数据移动，该命令允许功能用户控制其对软件的使用，而不是移动数据，例如向上/向下翻页命令、单击“确定”以确认错误消息等，请参阅第 3.6 节。

对任一兴趣对象所有数据属性的数据移动。

业务应用软件案例 4：最常见情况是对一个兴趣对象的所有数据属性的写操作，即：将该兴趣对象下所有数据属性的值保存到持久性存储介质中。

不同的功能用户移动同一兴趣对象的不同数据组的数据移动。

实时软件案例 2：一个功能处理被要求接收来自两个不同地震仪（功能用户）的数据组。这时，这两个地震仪对同一事件（比如:爆炸探测）的响应，应该被识别两个输入。

业务应用软件案例 5：假设 FUR 中某功能处理：对描述了同一兴趣对象但面向不同功能用户的不同数据组，识别多个输出。比如，当新员工入职时，公司会生成一份员工报告让员工签字确认个人信息有效，并向安保部门发送信息授权该员工进入办公楼。这里要识别两个输出。

数据移动：同一兴趣对象的不同数据组与持久性存储介质之间的数据移动。

业务应用软件案例 6：假设 FUR 中某功能处理 A 要存储从银行账户文件中提取的两个数据组，供其他的功能处理后续使用。第一个数据组是“透支账户详细信息”（包括了负余额的属性）。第二个数据组是“高价值账户的详细信息”（只包括了账户户主姓名和地址，用于寄送广告）。对于“账户”这个兴趣对象的不同数据组，需要识别出两个写。

业务应用软件案例 7：假设 FUR 中某功能处理是合并两个描述了同一兴趣对象的持久性数据文件，一个是关于兴趣对象“X”的现有数据文件，另一个是对该兴趣对象“X”的新增属性文件。在这个功能处理里，要识别两个读，每个文件各一个。

描述同一兴趣对象重复发生的数据移动。

业务应用软件案例 8：假设某 FUR 要求某功能处理“读”一个数据组，但是开发人员决定通过两个指令实现它，以便从处于不同位置的持久性存储介质中检索到描述同一兴趣对象的不同数据属性集，这应识别为一个读。

业务应用软件案例 9：假设 FUR 要求某个功能处理对一个文件读多次，这应识别一个读。

实时软件案例 3：假设某实时功能处理中，FUR 要求必须以固定的时间间隔从给定的功能用户（例如硬件设备）输入相同的数据组两次，以便测量处理期间的变化率。在 COSMIC 中，数据的两次移动被认为是同一输入的多次出现。对于此功能处理，只能识别为一个输入。请注意：这两次的输入都没有与之关联的数据运算，变化率的数据运算与输出关联。

实时软件案例 4：假设有一个生产纸张或塑料薄膜等扁平产品的机器的过程控制系统。该机器有 100 个相同的传感器方阵，用于监测产品上的裂纹。检查产品裂纹的功能处理需要从每个传感器中接收相同的数据。裂纹的位置信息可以通过传感器方阵发送的数据串来确定。所有传感器的数据处理流程都是一样的。在这个功能处理里，为所有传感器识别一个功能用户和一个输入。

当查询的输出是固定文本时的数据移动。

业务应用软件案例 10：按下“条款及条件”的按钮（比如在购物网页上），会输出一个固定文本，识别一个输出。

当功能处理需要从持久性存储介质移动一个数据组时的数据移动。

案例 1：这个案例涉及到一个软件块 A，需要检索一个存储的数据组，而软件 A 的 FUR 并不关心这些数据是如何被处理的（由处于同一层或不同层的其他软件）。

例如，如果软件 ***A*** 位于应用层，对一个已存储的数据组进行查询，那么软件 ***A*** 的功能用户可以是人类用户。图 ***3.2*** 展示了此查询的 ***COSMIC*** 数据移动。被一个输入触发的查询，紧跟着一个来自持久存储介质的数据组的读，然后是带有查询结果的输出。***功能处理 A*** 不关心从哪里检索的数据，只需要作为持久存储数据对待即可。

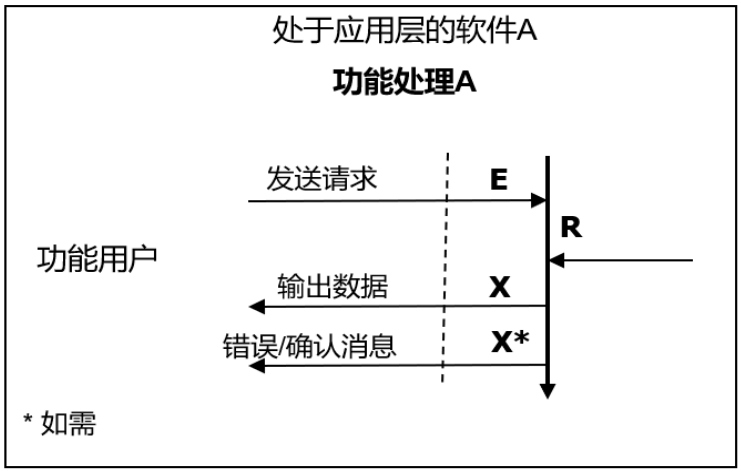

图 3.2 – 在应用层的软件 A 发送读操作的解决方案

如果需要功能处理 A 通过写数据移动把数据组移动到持久性存储介质中，该模型也同样适用。根据第 1 部分的规则 d)的错误消息，读和写数据移动都可能是报告错误消息的原因。

图 3.2 展示了功能处理 A 可能存在的错误消息，比如，所查询的记录未找到。然而，错误消息并不仅仅是软件特有的，比如“磁盘故障”，不可以计数为功能处理 A 的输出。请见第 1 部分 4.9 节的错误/确认消息。

当功能处理需要从其他软件块中获取数据时。

案例 2：待度量的软件块假设存在“客户端/服务器端”的关系，即存在客户端以接受处于同层或不同层的另一块“服务器”提供的服务和数据。图 3.3 是这种关系的一个例子，这两个组件是同一应用程序中的两个软件块。在所有的客户端/服务器端的关系中，客户端组件 C1 的 FUR 会把服务器组件 C2 识别为其功能用户，反之亦然。如果这两部分是独立的应用程序，或者其中一部分是独立的应用程序组件，关系也不会发生变化，关系图也是一样的。

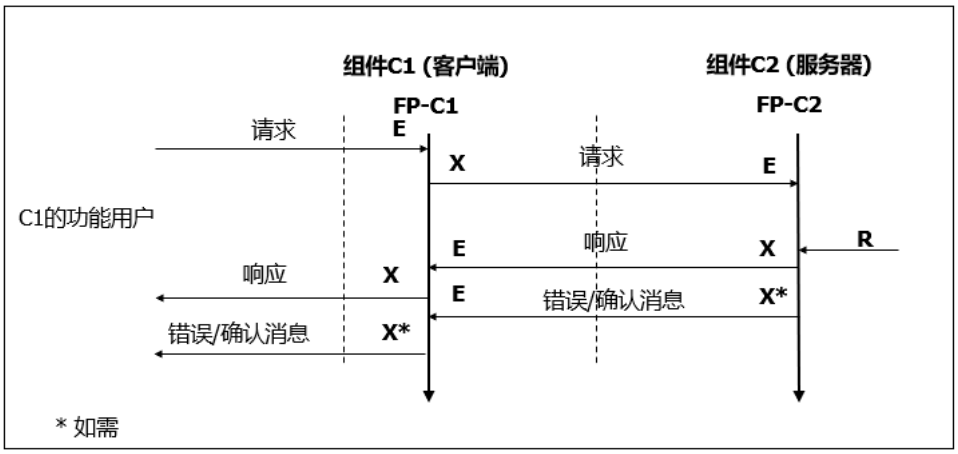

图 3.3 – 客户端-服务器端的数据交互

理论上，这两个组件可以在独立的处理器上执行，在这种情况下它们通过各自的操作系统以及其他中间层交换数据，软件架构如图 2.2 所示。但是逻辑上，转换为 COSMIC 模型后，这两个组件通过输入以及输出数据移动交换数据。所有中间的软件和硬件都被忽略了。

图 3.3 展示了客户端组件 C1 的功能处理 C1，是由功能用户（比如人）的输入所触发，输入可能包括查询的参数。C1 组件的 FUR 将识别该组件必须向服务器端组件 C2 索要数据，并且必须告知服务器端所需的数据组。

为了得到所需的数据组，功能处理 C1 向组件 C2 发送一个输出，包含了查询的参数。这个输出数据移动跨越了 C1 和 C2 之间的边界，并成为功能处理 C2 的触发输入。功能处理 C2 假设通过读取其自身的持久性存储介质得到了所需的数据组，并且作为输出发送给 C1。功能处理 C1 收到该数据，作为其输入。功能处理 C1 接着把该数据组作为输出发送给功能用户，以满足其查询需求。

考虑到客户端可能发出的错误/确认消息，这个案例 2 的查询功能处理因此可能需要 6 个数据移动，即 C1 是 6CFP，C2 是 4CFP。如果把 C1、C2 作为一个整体，可以通过读数据移动在其自己的边界内从持久性存储介质中读数据，则只需要 4CFP（1E,1R,2X）。

组件 C2 可能会使用存在于软件架构的其他层的存储设备驱动程序来读取硬件中的数据，如图 3.5（b）的案例 4。

数据移动和对存储数据的不同访问权限。

案例 3：见图 3.4，假设待度量的软件块 A 允许检索存储数据 Z，但是不允许维护（即创建、更新或删除）数据 Z。软件块 B 需要通过一致性验证确保数据 Z 的完整性，因此它处理数据 Z 的所有维护请求。当 A 需要维护数据 Z 时，A 必须通过输出，把请求传递给 B。

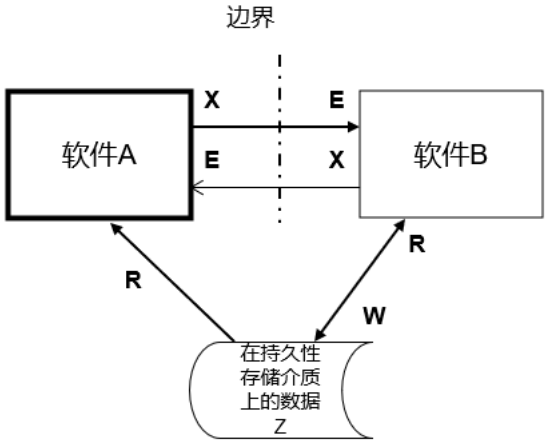

图 3.4 – 持久性存储数据 Z 在软件 A 和 B 的边界内读取。

当与物理存储设备交互的设备驱动软件读取持久性数据时的数据移动。

基础设施案例：本案例涉及到案例 1 中的需要检索一个存储数据组的软件块 A。考虑到独立的软件块 B 是智能硬件存储设备的设备驱动，在硬件存储设备中包含了软件 A 要获取的数据组。（简单起见，忽略操作系统的存在；操作系统可以把软件需求传递给设备驱动软件并且返回请求结果）。

两个软件块位于软件架构中的不同层，如图 2.2 所示，软件 A 位于应用层，软件 B 位于设备驱动层。物理上，在这两个软件块之间可能存在层次关系（忽略操作系统）以及一个物理接

口，如图 2.2 案例所示。然而，软件 A 和软件 B 的功能处理的模型本质上的关系是独立于各层之间的，这些关系可能是分层的也可能是双向的。

驱动层中软件 B 的功能用户是软件 A（忽略操作系统）和存有所需数据的智能硬件存储设备（“智能”指设备必须被告知需要的是什么数据）。

假设某软件 A 的查询功能处理 A 需要检索一个存储的数据组。图 3.5（a）展示了该查询功能处理的 COSMIC 模型。图 3.5（b）展示了在设备驱动层负责从硬件存储设备（如光盘或 USB）物理检索所需数据的软件 B 中的功能处理 B。

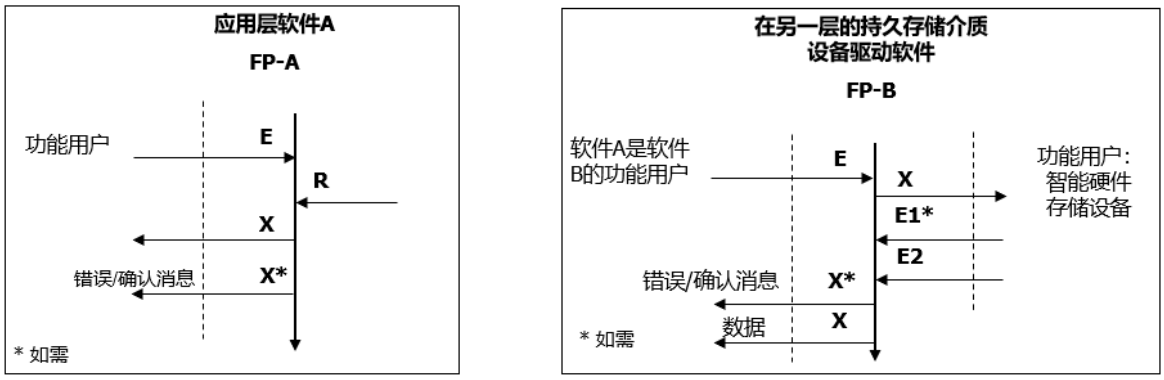

图 3.5 (a) 和 (b) – 应用层软件 A 向设备驱动层软件 B 发送读请求的解决方案

图 3.5（b）展示了软件 A 的读请求作为触发输入发送给功能处理 B，而后功能处理 B 把该请求作为输出发送给了硬件设备。后者的响应依赖于特定的硬件设备。设备可能仅仅返回请求的数据，如图 3.5b) E2。设备可能也会有错误消息，描述请求成功或请求失败的原因，比如“数据未找到”，或“磁盘错误”，如图 3.5b)中的 E1。功能处理 B 向软件 A 返回的值作为输出。功能处理 B 也可能发送常规的“返回码”，描述请求成功或请求失败的原因。（尽管返回码可能物理地附加在返回数据上，逻辑上该返回值是不同的数据组，因为它是关于请求过程的成果的数据）。对于功能处理 A，关于错误消息没有识别为输入，因为返回数据和错误消息都属于读数据移动，而相关的错误消息报告属于读和写。对于功能处理 A，识别一个输出作为错误/确认消息。

注意：实践中，在设备驱动软件和智能硬件设备间，除了如图 3.5b)所示之外，可能存在更多的数据移动。例如，此图没有显示设备驱动程序测量硬件无响应超时结果。

当功能处理不需要告知功能用户需要发送什么数据时的数据移动。

实时软件案例 5：假设某实时过程控制软件系统的功能处理需要轮询一组全部相同的哑传感器。在应用层，功能处理对数据的请求和数据的接收计数为 1 个输入。（因为所有的传感器都是相同的，因此只识别 1 个输入）。

进一步假设，对数据的请求必须传递给处于软件架构中较低层的设备驱动软件块，该驱动软件块理论上要从感应器方阵中获得数据，如图 2.3 中的层次架构所示。哑传感器的过程控制软件的功能处理和设备驱动软件的功能处理见图 3.6（a）和（b）。

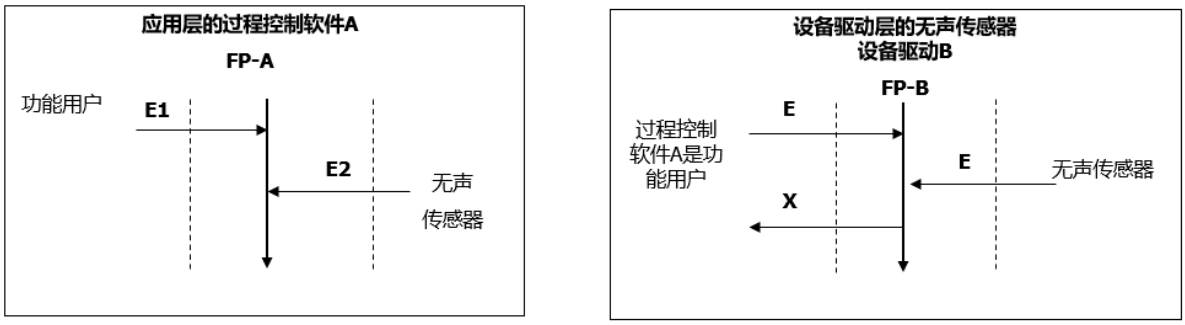

图 3.6 (a)和(b) －一组哑传感器的解决方案。

图 3.6（a）展示了软件功能处理 A 由输入 E1 触发，比如从时钟节拍。该功能处理接着通过 E2 从哑传感器方阵中获得数据，以接受传感器的多次读。哑传感器也是过程控制软件的功能用户（设备驱动软件隐藏在该层）。

图 3.6（b）展示了哑传感器驱动设备软件的模型。该驱动软件通过从过程控制软件（可能实际上是操作系统）发送的输入接收数据，作为功能处理 B 的触发。该功能处理通过从其功能用户（哑传感器）得到的输入 E 获得了所需的数据。

数据组通过一个输出被回传给过程控制软件。这个输出作为输入 *E2* 被功能处理 *FP A* 接收。*FP A* 随后继续处理传感器数据。同样地，从每个相同传感器收集数据是循环多次发生的，这不影响功能处理的规模。

来自处理控制应用程序软件的哑传感器的一个输入 *E2* 和来自设备驱动软件的被一个输出紧跟着的输入，这两者之间明显不匹配，因为按惯例，来自哑传感器的输入被认为包括任何“输入请求”功能（因为哑传感器功能用户没有处理任何来自功能处理的信息的能力）。

当功能用户需要被告知发送什么数据时的数据移动。

实时软件案例 6：假定功能处理发送一些查询参数或计算参数，或要压缩的数据到它的一个功能用户，如一个“智能”硬件设备或另一个对等软件块。功能用户的回应是通过一个功能处理发出的输出紧跟着一个输入数据移动，如 *3.4* 节案例 *2* 所述。

3.5 与数据移动关联的数据运算

数据运算被认为由相关的数据移动负责。对于由需求变更引起的必须被度量的数据运算请见 3.10。

业务应用软件案例 1：一个输入包括为使人类用户能够输入数据并确认所输入数据而对屏幕进行格式化的所有运算，但不包括确认某些输入的数据或代码，或获取相关代码描述而需要的读。

业务应用软件案例 2：一个输出包括为了格式化输出和为打印（或输出在屏幕上）准备某些数据属性而进行的所有运算，包括人类可读的标题栏[2]，但不包括为提供其中一些打印数据属性的值或描述而需要的读或写。

[2] 这个例子适用于度量人类用户使用的度量软件，与领域无关。显然，在度量复用对象的规模，比如在输入或输出屏幕上显示的单个字段表头，是不适用的。

案例：某功能处理的两个实例情况。假设在第一次发生时，由某个数据移动所移动的属性值导致数据运算子处理 A，而第二次发生时，导致另一个数据运算子处理 B。这种情况下，数据运算子处理 A 和 B 应该与同一个数据移动相关联，因此在该功能处理中只识别一个数据移动。

3.6 控制命令

在 COSMIC 方法中，控制命令不属于数据移动。识别控制命令以避免错误识别是很重要的。

控制命令的案例。

- 在物理屏幕中上下翻页。
- 按 Tab 键或回车键，或按某个按钮继续。
- 点击“OK”按钮，确认或取消先前的操作，知晓错误消息，或确认输入数据等。
- 允许用户控制展示或隐藏已完成计算的小计的表头。
- 菜单命令，允许用户浏览一个或多个功能处理，但是并不会启动它们。
- 展示空屏幕待数据输入的命令。

3.7 错误/确认消息和其他出错状态的提示

错误和确认消息，如果有的话，是度量的对象。

案例：错误/确认消息可以是：输入数据验证成功或失败的消息，或者用于检索数据或存储数据，或者响应来自另一个软件块的服务。

业务应用软件案例 1：在人机交互的场景下，在验证输入数据时发生的错误消息可能有“格式错误”，“用户未找到”，“错误：请勾选方框表明已阅读条款”，“超出额度上限”等。在每个功能处理中，所有这些错误消息应该只识别为一个输出，

业务应用软件案例 2：功能处理 A 可以向其功能用户发出 2 个不同的确认消息，和 5 个错误消息。对这一共 7 个错误/确认消息只识别 1 个输出。功能处理 B 可以向其功能用户发送 8 个错误消息。对这 8 个错误消息识别 1 个输出。

具有额外数据的错误/确认消息。

业务应用软件案例 3：银行 *ATM* 机（即自动提款机）的某功能处理，当提取特定金额的现金时，会有以下五类消息提示：

- 错误：机器中无现金。
- 错误：输入的金额必须为$10 的倍数。
- 提取现金失败。账户被锁定，请联系银行。
- 取钱失败（超出余额$139.14）。
- 取钱成功，账户余额$756.25。

前四个消息描述了错误消息，第五个消息的前半部分是一个确认消息。根据识别错误/确认消息的规则，这五个消息先都识别为 1 个输出。最后两条信息还包括了与用户账户相关的数据属性，为该数据再多识别 1 个输出，因为涉及了客户账户数据的移动。总的来说，为该功能处理识别 2 个输出，即 2CFP。

没有在 FUR 中说明的发送给人类用户的错误消息。

业务应用软件案例 4：从操作系统发来的消息比如“某打印机未响应”，忽略该类消息。

实时软件案例 1：在实时软件系统中，周期性检查所有硬件设备的功能是否运转正常的功能处理可能会发送这样一条消息，报告传感器 S 失灵，S 是变量。这条消息应该被识别为 1 个输出，因为引发了功能用户（兴趣对象）传感器 S 的数据。

表示了错误信息和其他数据的情况。

实时软件案例 2：3.4 节的实时软件案例 6 也可以这样描述，当设备驱动软件无法从哑传感器方阵中获得数据时，功能处理 A 和 B 必须发出错误消息。因为根据定义，哑传感器无法发出错误消息。设备驱动功能处理 B 将很有可能会从哑传感器方阵中得到一串值，比如状态 1，状态 2，状态 3，无响应，状态 5，无响应，状态 7 等等，并且把这串值作为输出发送给应用软件的功能处理 A，作为其输入。对于设备驱动软件的功能处理 B 的输出和过程控制软件的功能处理 A 的输入都不单独识别错误消息。

3.8 度量分布式软件系统的构件

请见 3.4 节的例子 2。

3.9 软件的复用

包含了通用功能的功能处理。

业务应用软件案例 1：待度量软件中的多个功能处理可能需要相同的验证功能，比如“订单日期”，或可能需要访问相同的持久性数据，或可能需要执行相同的运算。在每个功能处理中，按需度量通用功能。

实时软件案例 1：待度量软件中的多个功能处理可能需要获得来自同一个传感器的数据（移动同一个数据组），或是需要进行相同的比例换算，比如从华氏度转为摄氏度（相同的数据运算）。像上一个例子一样按需度量该通用功能。

不在 FUR 中的功能。

业务应用软件案例 2：对于业务应用软件来说，启动某程序的用户可能是操作系统中的时间表组件，或是电脑操作员，或是其他人类用户（比如 PC 用户启动浏览器或文辞处理软件时）。这些用户所传递的数据没有写在 FUR 中也不会被应用软件处理，因此忽略。

实时软件案例 2：：对于实时软件，启动某程序的用户可能是操作系统或网络管理生成的时钟信号，或是人类操作员（比如从操作台启动过程控制系统），这些都必须忽略。

基础设施软件案例：对于计算机操作系统来说，启动操作系统的用户是一个引导程序，当计算机电源打开时启动，这个也必须忽略。

业务应用软件案例 3：以批处理模式为各种各样的功能处理输入数据的应用程序可能是由操作系统的一个调度程序启动。如果目的是度量批处理应用程序的 *FUR*，应该忽略“启动系统”这一功能。批处理应用程序的功能处理的触发输入和其他所需输入将组成该批处理应用程序的输入数据。

业务应用软件案例 4：在特殊情况下，在每个时间段结尾生成摘要报告的批处理软件可能不需要任何直接来自功能用户的输入数据便可以启动。关于该案例的分析请参考 3.1 节的业务应用案例 7。

实时软件案例 3：现代化的汽车具备分布式的电子控制单元（ECU）系统以控制诸多功能，比如发动机管理，车闸，空调等。在 AUTOSAR 架构中，在一个分布式的系统里，“网络管理”模块总是在运行，该模块负责激活通过网络（总线）连接到一起的 ECU。该网络管理模块也负责转换 ECU 的操作状态：正常运行，电量低或休眠。因此，是网络管理把 ECU 唤醒或让其休眠。当度量 ECU 软件时，应该忽略网络管理功能。

3.10 度量软件的变更规模

COSMIC 方法的变更规模包括了所有影响规模的变更，也包括关联在数据移动上的数据运算。

案例 1：一个数据运算通过以下方式被修改：改变计算方法、特定的格式化、显示方式，和/或数据的确认。“显示方式”包括如字体、背景颜色、字段长度、字段标题和小数位数等等。

案例 2：假设某功能处理的变更需要对关联到其触发输入的数据运算进行三处变更，关联到其输出的数据运算进行两处变更，以及针对其输出的数据组的属性进行两处变更。度量该变更的规模是 2CFP。只考虑数据移动的属性及相关的数据运算的变化，而不考虑变更的数据运算或数据属性的个数。

案例 3：假设需要添加或修改某数据组 D1 的数据属性，修改后的数据组命名为 D2。在功能处理 A 中，需要进行上述修改，所有被修改影响的数据移动应该被识别和计数。因此，如果变更的数据组 D2 需要被存储到持久性存储介质和/或被输出时，需要识别一个写和/或一个输出数据移动。

如果其他功能处理对该数据组 D2 进行读或输入的数据移动，但是他们的功能并没有被修改所影响，因为他们没有处理变更的数据属性。这些功能处理将按照原样像处理 D1 一样处理。因此，不会被功能处理 A 中的数据移动的修改所影响的其他功能处理中的数据移动，一定不要识别和计数。

案例 4：通常，删除（用“分离”这个词可能更恰当）一个应用软件的淘汰部分，只是删除对它的调用，而代码仍然保留在那里。假设淘汰部分的功能总计为 100CFP，而这部分可以通过 2 个数据移动的变更被分离。变更的规模取决于度量目的。如果度量目的是把规模作为项目估算的输入，则变更规模是 2CFP。如果目的是删除那部分的规模，则变更规模是 100CFP。

出于估算目的，建议对变更的部分单独度量其生产力，因为切断连接和真正删除是有很大区别的。或者，为了估算目的，最好度量实现的规模而不是需求的规模。

案例 5：前一个案例，如果度量的变更规模是 2CFP，需要在文档中清楚记录，并且与 FUR 的度量区分，因为在 FUR 中是明确表示要删除这部分规模为 100CFP 的程序。

业务应用软件案例 1：如果错误/确认消息需要变更（比如增加、删除或修改文本），无论更改的文本是否由更改其他数据移动引起的，都应该进行识别。

对控制命令和程序通用数据的变更。

业务应用软件案例 2：对屏幕上的颜色的变更，不应该被度量。

3.11 COSMIC 度量方法的扩展

在 FUR 中无法使用规则度量的部分，虽然不是 COSMIC 方法的正式组成部分，但可能需要一个单独的规模度量流程。该规模不应包括在度量报告中。

案例 1：可以对 COSMIC 方法进行扩展，以度量 FUR 中的数据运算的规模。

案例 2：可以对 COSMIC 方法进行扩展，以单独度量每个数据移动的数据组个数对于软件规模的影响。