Measures

全功能点估计方法

COSMIC FFP (Common Software Measurement International Consortium full function point)

报告人

- 任甲林
  - 高级咨询顾问，主要从事提升软件研发能力的培训与咨询
  - 联系方式：
    - Mobile：15863181188
    - E-mail：renjialin@measures.net.cn
    - MSN: dylan_ren@msn.com
  - Blog：
    - http://dylan1971.blog.ccidnet.com/
  - 工程经验：
    - 93年从事软件开发, 参与了50多个项目的开发
    - 曾为北京汉王、上海鹏开、深圳富士康、深圳鹏开、四川长虹、大连华信、成都虹微、昆山中创、北京信城通、南京诚迈、南京润和、珠海矩力等多家公司咨询

内容

Measures

- 规模估算的意义
- 发展历史
- 适用的领域
- 基本思想
- 基本过程
- 度量策略
- 映射阶段
- 度量阶段
- 简单案例
- 完整案例

为什么要进行规模估计？

Measures

- 如果要你估计某人的体重，你会如何估计？
- 如果要你估计某个项目的工作量，你会如何估计？
- 如果有2个项目，项目A测试出了200个bugs，项目B测试出了100个bugs，哪个项目质量更差呢？

基于代码行与功能点的的规模估算方法

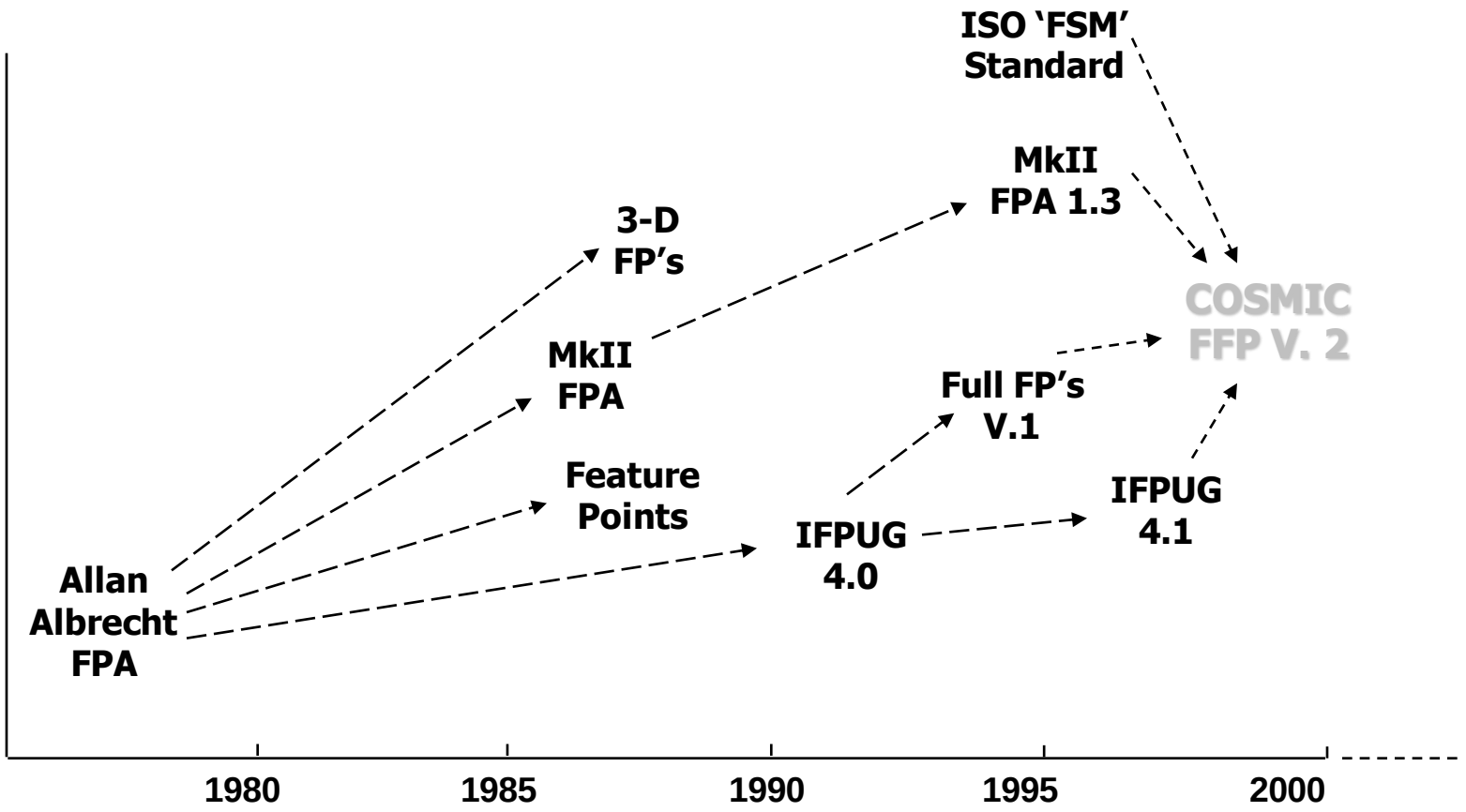
代码行

- 优点
  - 人员不需要专门的培训，简单易行
  - 结果直观
- 缺点
  - 估计结果依赖于个人的估算水平
  - 同一个系统不同的技术平台下估算出规模不同
  - 不同的人员实现同一个系统，实际规模可能相差很大
  - 不同的系统之间无法比较规模的大小
  - 需要明确定义代码行的含义
    - 是否含有注释行
    - 是否含有空行
    - 是逻辑行还是物理行
    - 是否包含开发平台自动生成的代码
    -

功能点

- 优点
  - 与技术平台无关
  - 与实现的人员无关
  - 系统之间的规模具有可比性
  - 客观度量，不依赖于参与估算的人员
    - 通过认证的功能点分析师一般能产生彼此相差不超过10%的计数值
- 缺点
  - 估算与度量人员需要经过专门的培训
  - 估算结果不够直观

发展的历史


ISO 'FSM'
Standard
MkII
FPA 1.3
3-D
FP's
COSMIC
FFP V. 2
MkII
FPA
Full FP's
V.1
Feature
Points
IFPUG
4.1
IFPUG
4.0
Allan
Albrecht
FPA
1980
1985
1990
1995
2000

ISO标准

- 国际功能点用户协会（International Function Point Users' Group，IFPUG）（www.ifpug. org）提出的IFPUG功能点分析方法，相应的国际标准编号是ISO/IEC 20926:2003。
- 荷兰软件度量协会（NEtherlands Software Metrics Association，NESMA）（www.nesma.nl）提出的荷兰软件功能点分析方法，相应的国际标准编号是ISO/IEC 24570。
- 英国软件度量协会（UK Software Metrics Association，UKSMA）（www.uksma.co.uk）提出的Mk II功能点分析方法，相应的国际标准编号是 ISO/IEC 20968:2002 。
- 通用软件度量国际协会（COmmon Software Measurement International Consortium ，COSMIC）（www.cosmicon.com ）提出的全功能点分析方法，相应的国际标准编号是ISO/IEC 19761:2003。

FFP功能点估算方法的简单示例

Measures

- 登录界面的功能点识别
  - 功能处理：登录
    - 输入 用户信息 1 CFP
    - 读 用户的密码信息 1 CFP
    - 输出 错误提示 1 CFP
    - 写 登录日志 1 CFP
  - 合计：4个功能点

功能点估算的经验法则

Measures

- 功能点数/150=软件开发人员数量
- 功能点数/3500=维护程序员数量
- 功能点数*1000美元=美国的软件开发成本
- 功能点数*2%=每月的需求蔓延率
- （功能点数）$^{1.2}$=软件中潜在的缺陷总数
- （功能点数）$^{1.25}$=需要的测试用例总数
- （功能点数）$^{1.15}$=全部文本文档的页数

FFP适用的领域

Measures

适用的领域

- 企业应用软件，大数据量处理为主
  - 银行、财务、保险、个人、采购、分销、制造
- 实时系统
  - 电话交换系统
  - 嵌入式控制软件
    - 家电中的控制软件
    - 汽车中的控制软件
    - 过程控制,自动数据采集系统
  - 操作系统
- 混合型
  - 实时的机票预定或旅馆预定系统

不适用的领域

- 复杂算法的系统
  - 专家系统
  - 仿真系统
  - 自学习系统
  - 天气预报系统
- 处理连续变量的系统
  - 声音和图象处理系统
- 对于这类的软件，可以在FFP方法基础上进行扩展、本地化。

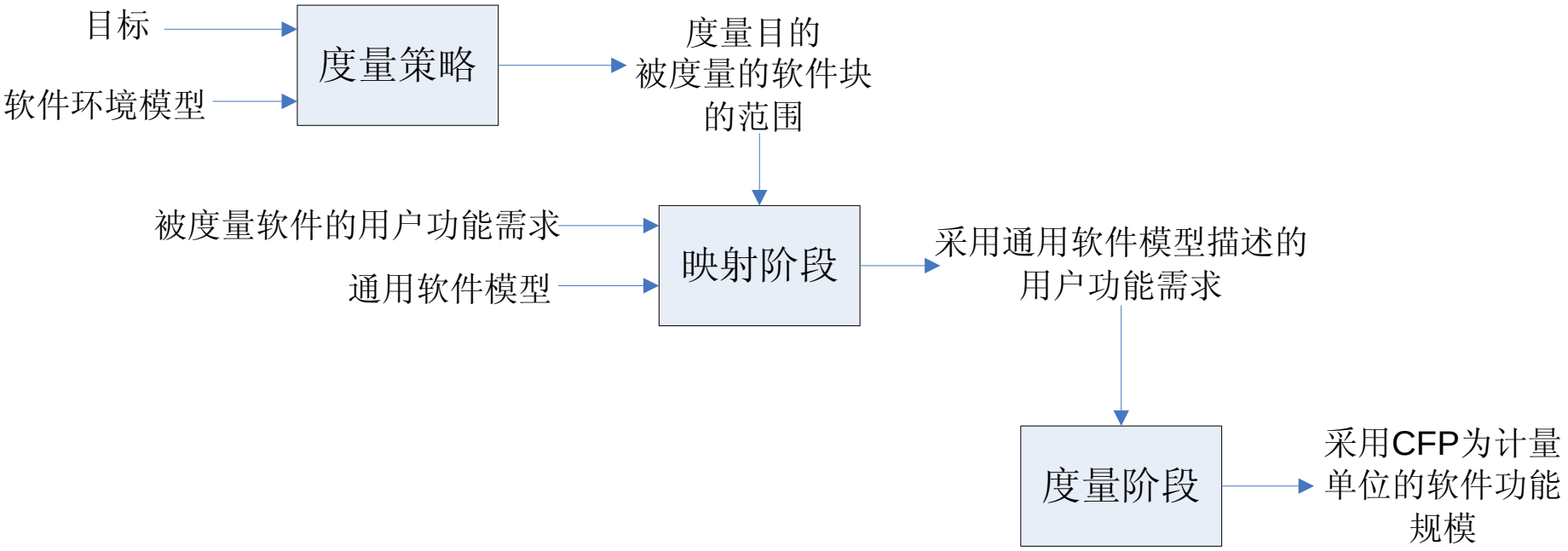
COSMIC FFP的限制因素

- 没有考虑：
  - 复杂度
  - 数据组中数据属性的多少
- 对于软件中的很小功能可能不合适

COSMIC-FFP的特点

- COSMIC-FFP 是一种功能规模度量方法,侧重于用户功能的视角
- 是一种适合于 MIS,实时系统和多层系统的简单模型
- 可以在软件开发生命周期的各个阶段使用
- 起源于客户可以理解的术语
- 不需要参考:
  - ✓ 工作量
  - ✓ 使用的技术方法
  - ✓ 物理的或技术的构件
- 不需要调整因子
- 不依赖于参与估算的人员
  - 通过认证的功能点计数师一般能产生彼此相差不超过10%的计数值

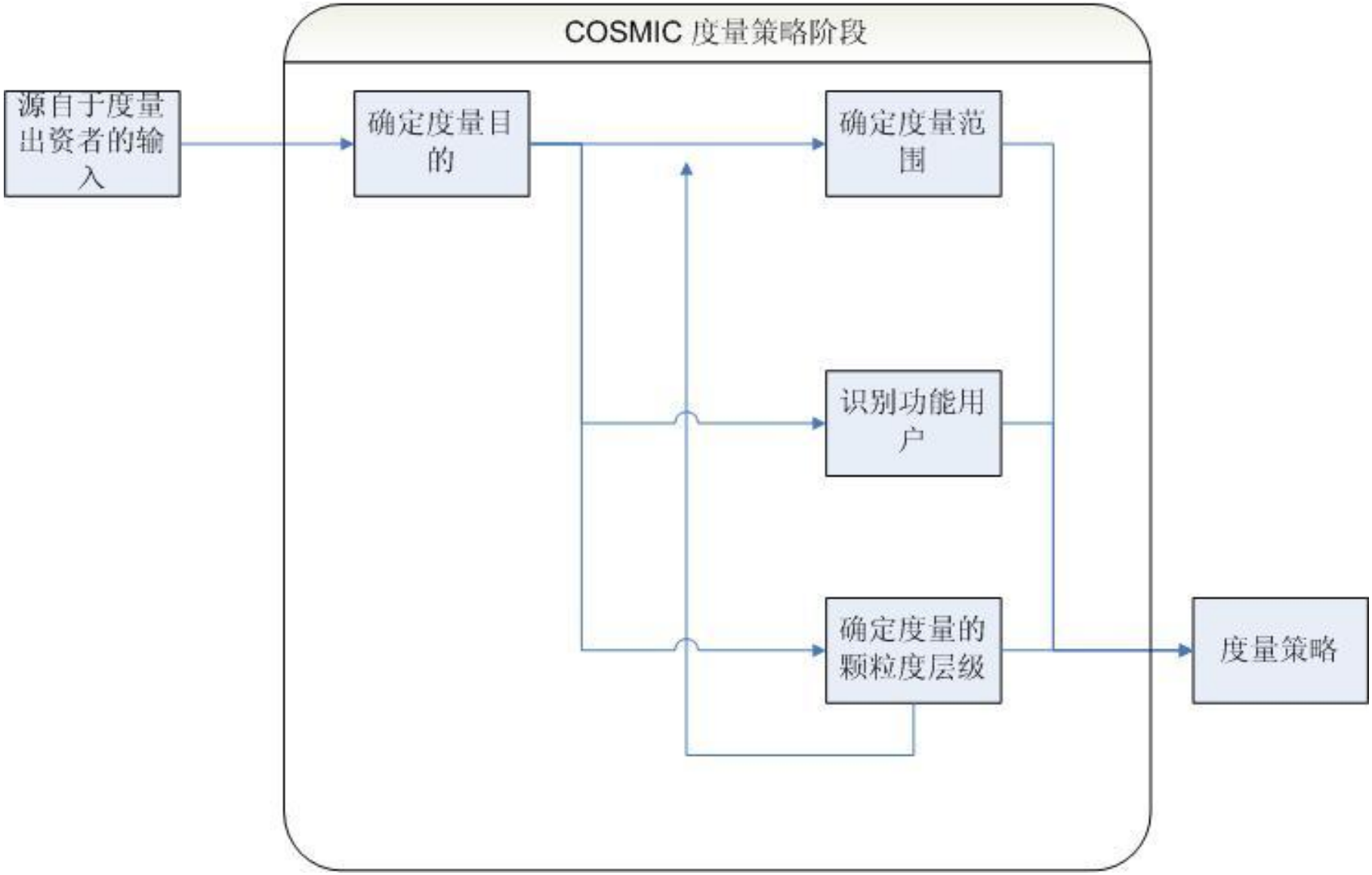
FFP的过程模型


目标
软件环境模型
度量策略
度量目的
被度量的软件块
的范围
被度量软件的用户功能需求
通用软件模型
映射阶段
采用通用软件模型描述的
用户功能需求
度量阶段
采用CFP为计量
单位的软件功能
规模

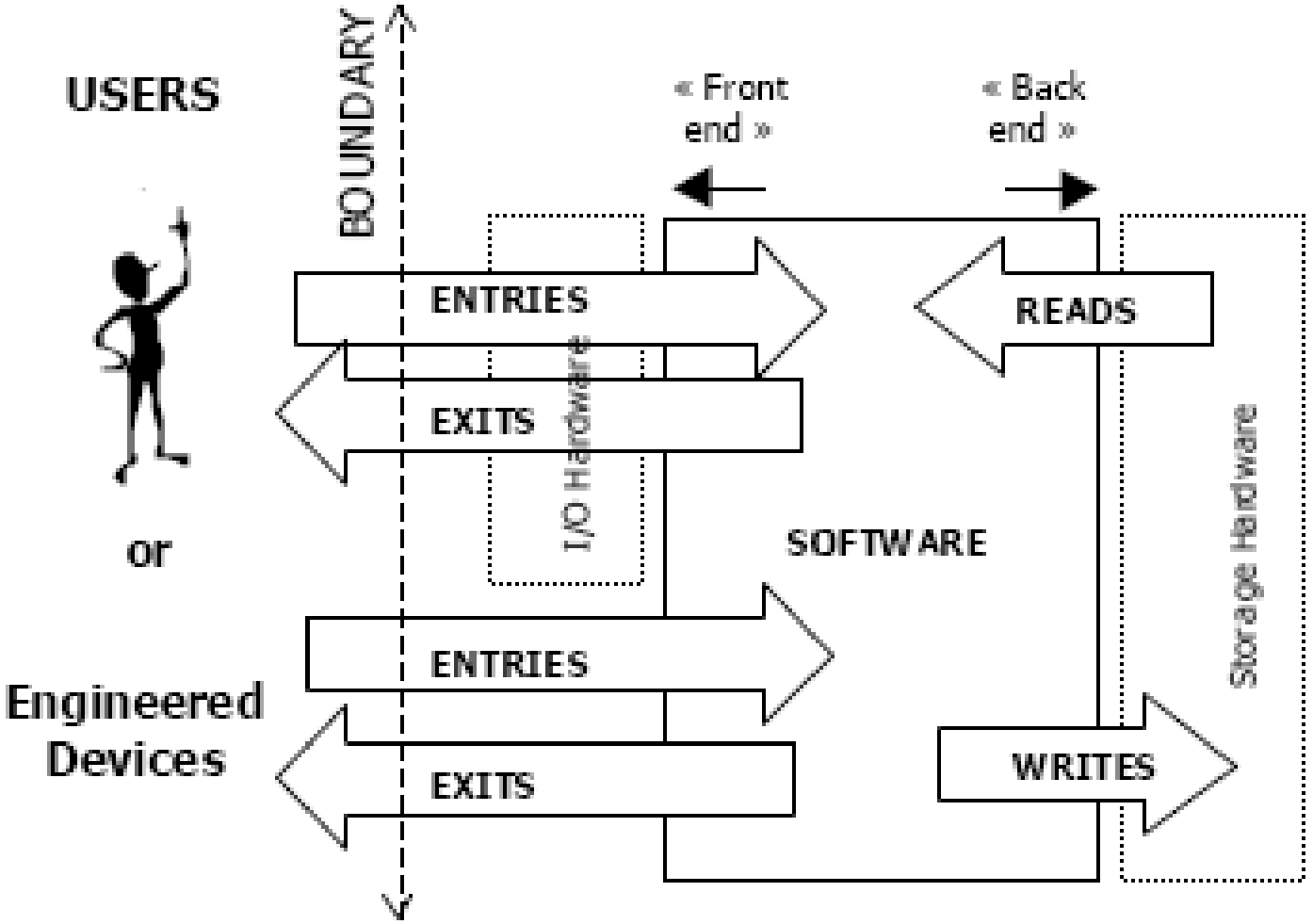
Measures

度量策略阶段

度量策略阶段


COSMIC 度量策略阶段
源自于度量出资者的输入
确定度量目的
确定度量范围
识别功能用户
确定度量的颗粒度层级
度量策略

软件环境模型


USERS
BOUNDARY
« Front end »
« Back end »
ENTRIES
READS
EXITS
I/O Hardware
SOFTWARE
Storage Hardware
or
ENTRIES
Engineered Devices
EXITS
WRITES

识别COSMIC软件环境模型的原则

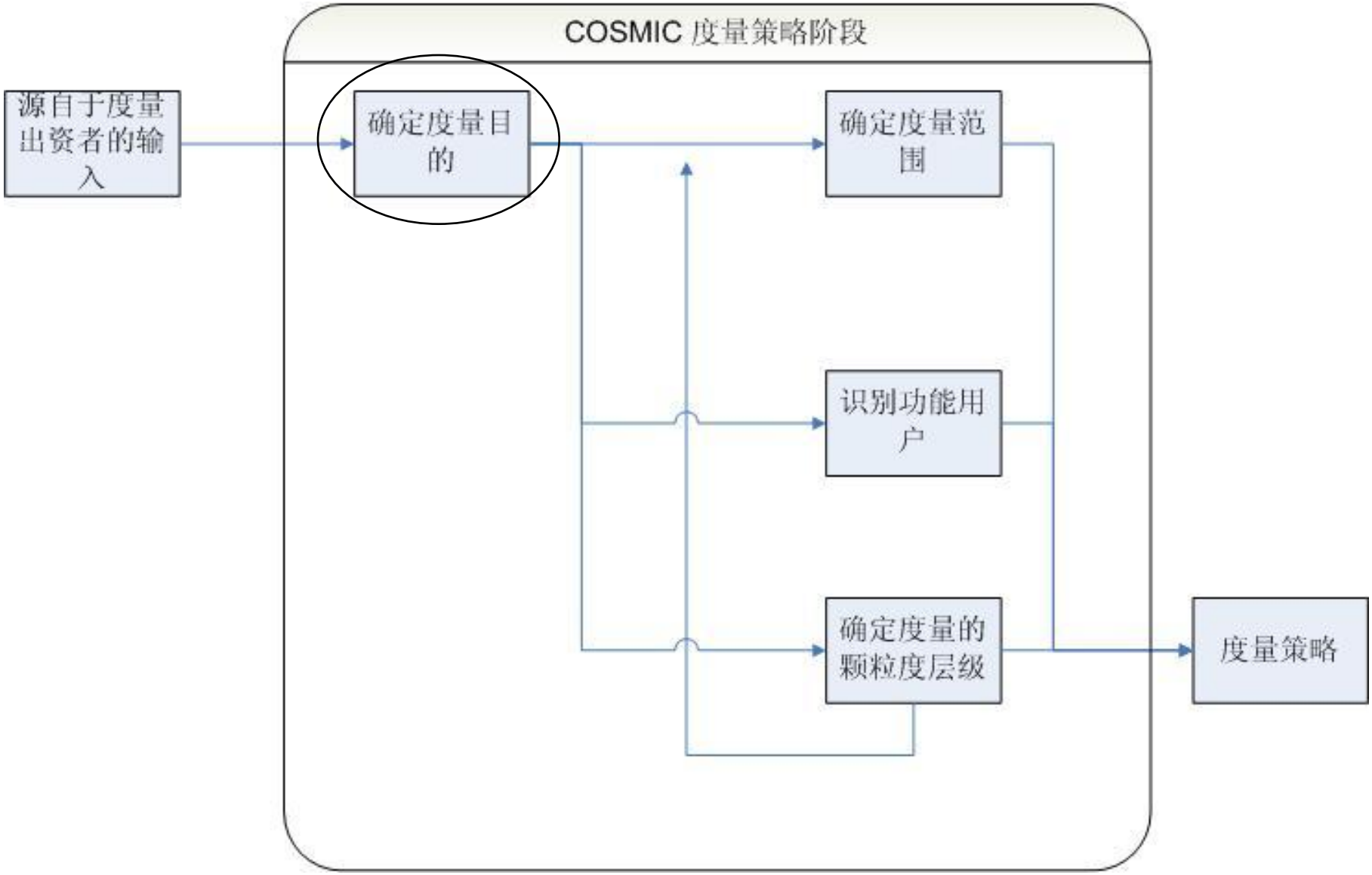
- a) 软件被硬件界定
- b) 软件通常结构化为多**层**
- c) 一个层包含一个或多个单独的**对等**软件块，并且任何一个软件块可能由独立的对等构件构成。
- d) 任何待度量的软件块可以由其**度量范围**定义，软件块应被完全限制在一个单一的层中。
- e)待度量的软件块的范围依赖于**度量目的**。
- f) 可以从待度量软件块的**用户功能需求**中识别该软件块的**功能用户**，这些功能用户作为数据的发送者和/或接受者。
- g)一个软件块通过穿越**边界**的**数据移动**与功能用户交互，并且，软件块在边界内向**持久存储介质**输入或者读取数据
- h) 软件的用户功能需求可以在不同**颗粒度层级**来阐述
- i) 度量规模时，软件的颗粒度通常是处在由**功能处理**所构成的那个层级上
- j)如果不能在功能处理的颗粒度层级进行度量，那么软件的用户功能需求应该通过近似的方法来度量，并且按比例缩放到功能处理的颗粒度层级

度量的四个主要参数

Measures

- 度量目的
- 度量范围
- 识别的功能用户
- 度量颗粒度的层级

度量策略阶段


COSMIC 度量策略阶段
源自于度量出资者的输入
确定度量目的
确定度量范围
识别功能用户
确定度量的颗粒度层级
度量策略

确定度量目的

Measures

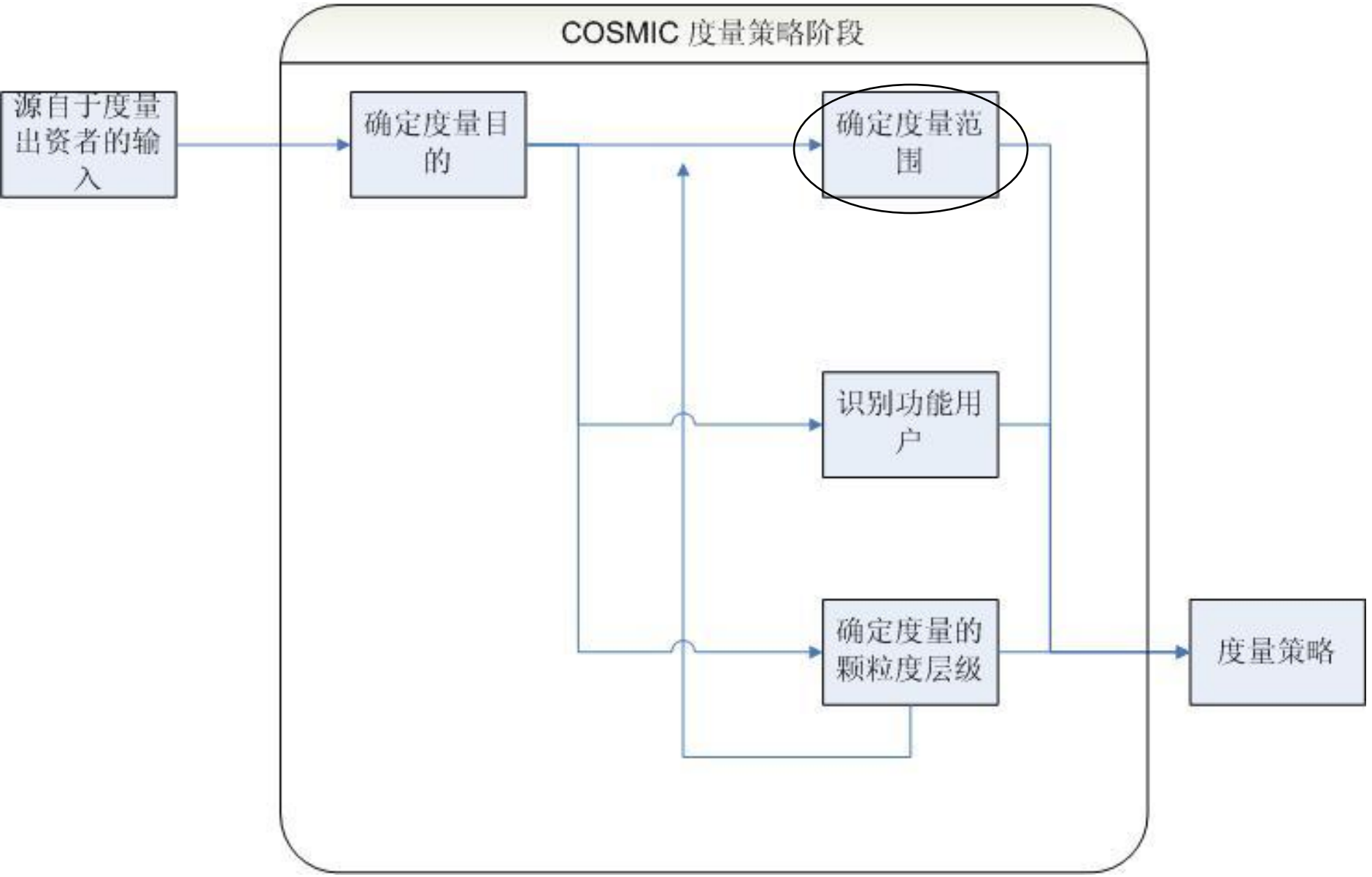
- 度量目的描述了为什么要度量，度量结果有何用途？
- 度量目的如：
  - 估算：
    - 随着功能需求的演变，度量其规模作为估算开发工作量过程的输入
  - 管理：
    - 在功能需求已经被认可后，度量其变化的规模，以管理项目范围的蔓延
  - 考核：
    - 度量已交付软件的功能需求的规模作为度量开发人员业绩的输入
    - 度量已有软件的功能需求的规模作为度量维护人员和支持人员的业绩时的输入
  - 了解现状
    - 度量已有软件的功能需求变化的规模作为一个维护组的产出的规模度量
    - 度量已有软件提供给操作人员的功能规模
  - 技术有效性：
    - 度量所有已交付软件的功能需求的规模，以及所有已开发软件的功能需求的规模，以得到复用功能的度量数据。

度量目的的作用

Measures

- 确定度量范围以及度量需要产出的工作产品
- 确定功能用户
- 确定在项目的生命周期中实施度量的时间点
- 确定度量的精度以及因此确定是否可以使用COSMIC度量方法或者是否使用一个本地化的近似的COSMIC方法。后面的2点将决定FUR的颗粒度层级。

度量策略阶段


COSMIC 度量策略阶段
源自于度量出资者的输入
确定度量目的
确定度量范围
识别功能用户
确定度量的颗粒度层级
度量策略

度量范围的定义

Measures

- 在某次特定功能规模度量中包含的用户功能需求的集合

- NOTE：“总体范围”与“范围”的含义不同：
  - “总体范围”是基于度量目的确定的所有软件
  - “范围”是指在总体范围内的需要分别度量的软件块

- 案例：
  - 度量目的：度量某销售管理系统的规模
  - 总体范围：销售管理系统
  - 范围：在该系统中包含了10个用例，每个用例需要分别度量其规模，然后累计起来得到总的规模。每个用例就是一个范围。

用户功能需求的定义

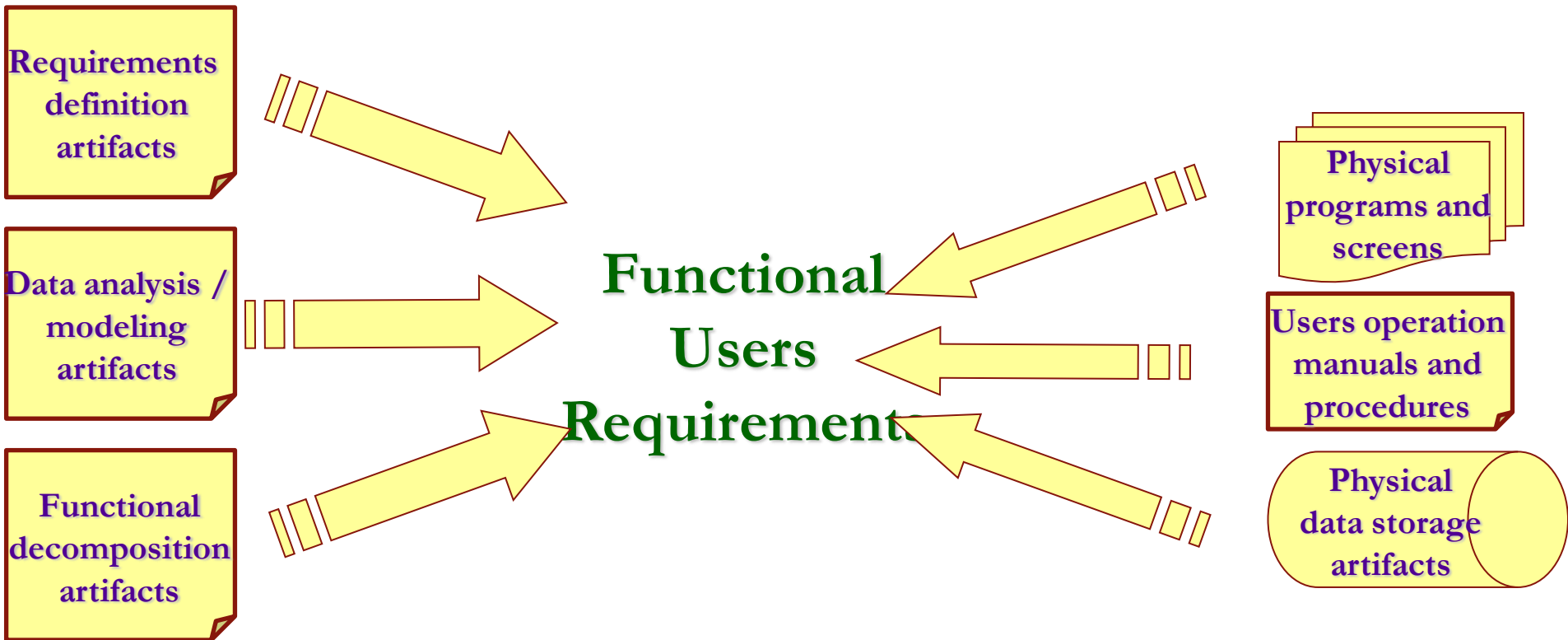
Measures

- 用户需求的一个子集。以任务和服务的形式描述软件做什么。
- 用户功能需求描述了为客户必须做什么而不包括任何描述了”怎么做”的技术和质量的需求
- NOTE:
  - 用户功能需求的例子包括但不限于如下方面:
    - 数据传送(例如，输入客户数据；发送控制信号)
    - 数据转换(例如，计算银行利息；计算出平均温度)
    - 数据存储(例如，存储客户订单；随着时间流逝，记录周围的温度)
    - 数据检索(例如，列出当前职员；获取飞机当前位置)
  - 是用户需求但不是用户功能需求的例子包括但不限于如下的:
    - 质量约束(例如，可用性、可靠性、效率和可移植性)
    - 组织级约束(例如，操作位置，目标硬件和与标准的一致性)
    - 环境约束(例如兼容性、安全性)
    - 实施限制(例如，开发语言，交付进度)

用户功能需求

Measures

- 可以在软件开发出来之前从软件工程的文档中抽取出来。
- **FUR也可以从软件开发开发完成之后的工作产品中抽取出来**
- FUR可能清晰，也可能模糊，在估算时，可能需要估算者做出推理或假设。



规则-范围

- a） 功能规模度量(FSM)的范围应该派生自度量目的。
- b） 任何一次度量的范围都不应该超过被度量软件的一个层次。

- 对比建造房屋，如果度量的目的是成本估计的话，那么就很有必要单独度量房屋的不同的部分了，比如，房屋的地基、墙和屋顶，因为他们使用了不同的建造方法。不同的技术可能对应了不同的生产率指标。

度量目的决定了度量范围

Measures

- 案例：
  - 目的1：度量一个项目组交付的软件的规模，则其范围应包括：
    - 最终用户日常应用的模块
    - 系统实施人员只使用一次的数据升级模块
  - 目的2：度量新系统运行后，用户使用的软件规模，则其范围应包括：
    - 最终用户日常应用的模块
    - 不包括数据升级的模块，系统的范围小了，规模小了
- 综上所述，度量目的用于判断
  - （a）哪个软件被包含在整个范围之内或者哪个被排除与整个范围之外
  - （b）被包含在内的软件需要被分成单独的小块，每一个小块具有其自己的范围，单独进行度量

定义—分解的层级

Measures

- 对软件块的任何分解展示的都是其构件、子构件等等
- 举例：
  - 应用软件包
    - 应用软件1
      - 主要构件
        - » 可复用的对象类
    - 应用软件2
      - ….
    - ……
- 注意：
  - 不要与“颗粒度级别”混淆
  - 在比较不同构件的规模大小时应在分解的不同层次上进行比较
  - 任何软件块的规模不能仅仅通过合计其构件的规模而得到

层次的定义

Measures

- 层是软件架构按照功能分解而形成的一个部分，层与硬件形成整个计算机系统：
  - 层是按照层次结构来组织的
  - 在层次结构的任何一个级别，仅仅有一个层
  - 在软件架构的任意两层提供的功能服务之间存在一种可以直接交换数据的“主/从”层次依赖关系
  - 在软件架构中交换数据的任意两层软件仅部分相同的解释交换的数据

识别层次的原则

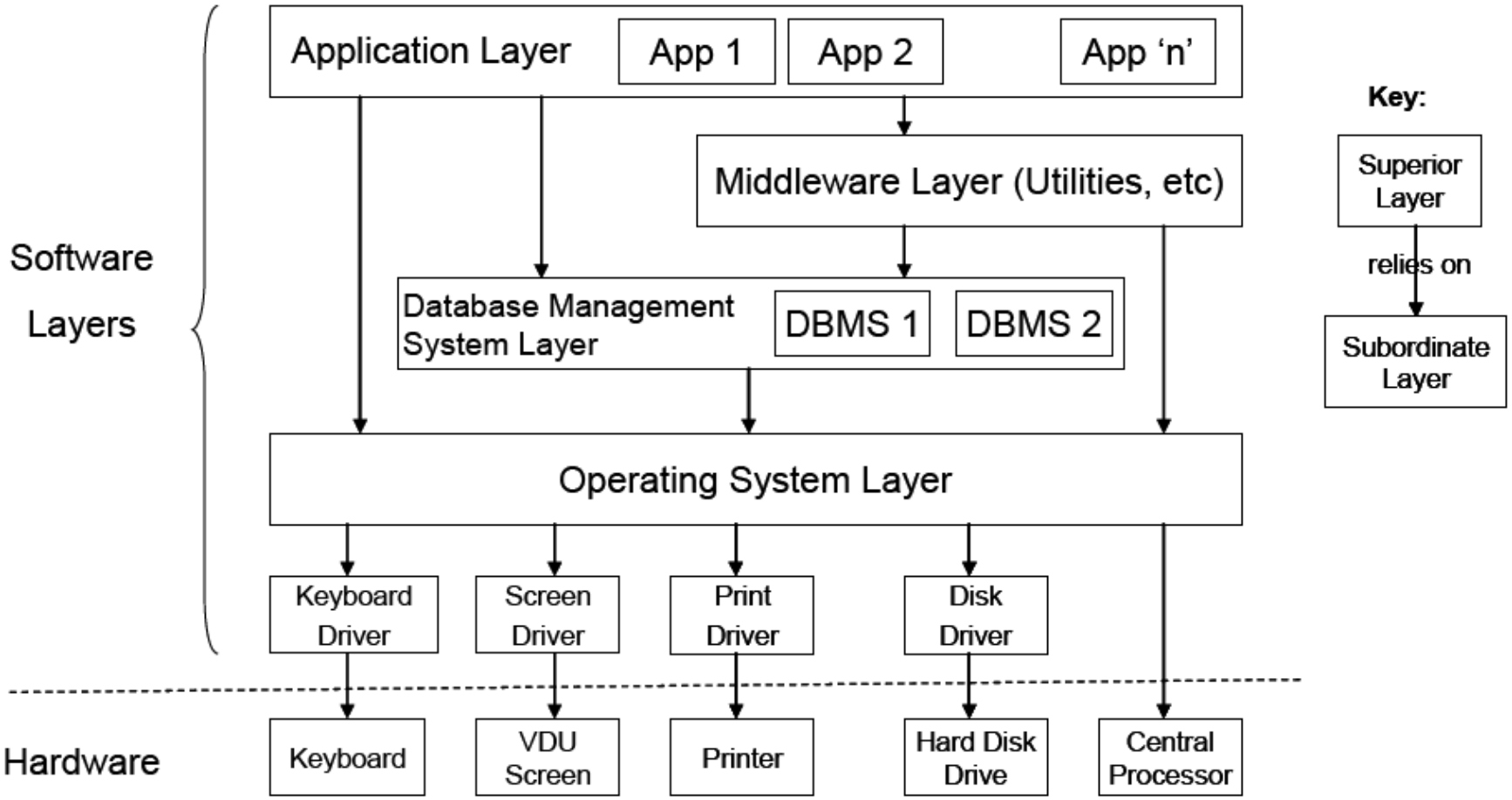
Measures

- 一个层的软件与其他层的软件通过其各自的功能处理交换数据
- 层之间的“层次依赖”就是指任何一个层的软件可以利用任何其下层的功能服务。 存在此类使用关系的时候，我们把使用软件层看作“主层”，任何包含被使用的软件的层叫做“从属层”。主层软件依赖于从属层软件的服务才可以正常运行;从属层依次依赖于他们的从属层才可以正常运行，沿着层次体系依此类推. 相反的，从属层的软件和它所依赖的所有从属层的软件可以不需要任何主层的服务而运行。
- 一个层中的软件不必使用所有的从属层的软件提供的功能服务
- 任意两层的软件之间交换的数据在这两个软件块的FUR内部可以分别进行不同的定义和解释，也就是说着两个软件块可以对交换的数据识别为不同的数据属性和/或数据子组，但是，根据接收方软件的需求，一定存在一个或多个共同的数据属性或数组子组使接收层的软件能够解释传送层发送的数据。

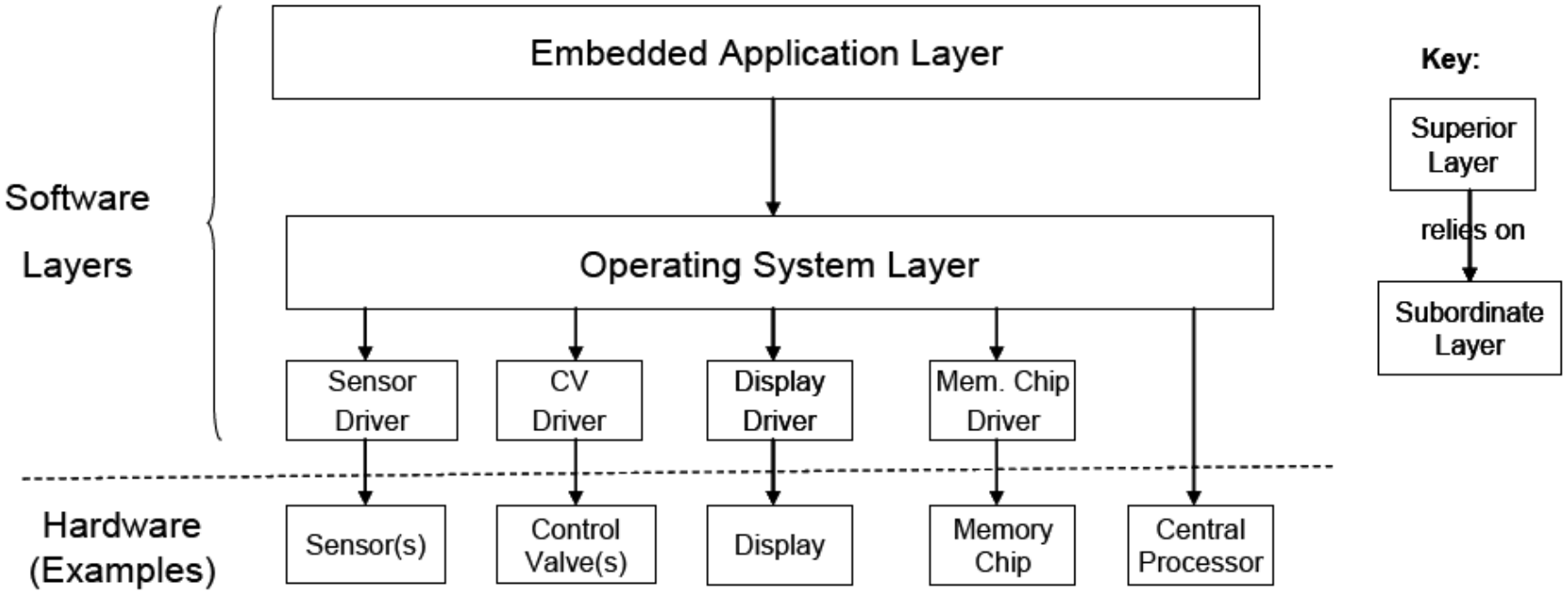
识别层次的规则

- a)如果软件是采用一种本模型里的层次体系结来表达的，那么这种架构就可被用来为度量目的识别层次.
- b)在MIS或商业软件领域，顶层，例如，该层不是其他任何一层的从属层，通常叫做“应用层” 。该层中的(应用)软件依赖于其它所有层软件的服务。在实时软件领域，顶层软件通常是指一个“系统”，例如，“过程控制系统软件”，“航班控制系统软件”
- c)不要假设任何没有进行层次结构设计或结构化考虑的软件可以按照COSMIC模型进行分层

对于商业应用软件的典型的分层结构



对于实时嵌入式系统典型的分层结构



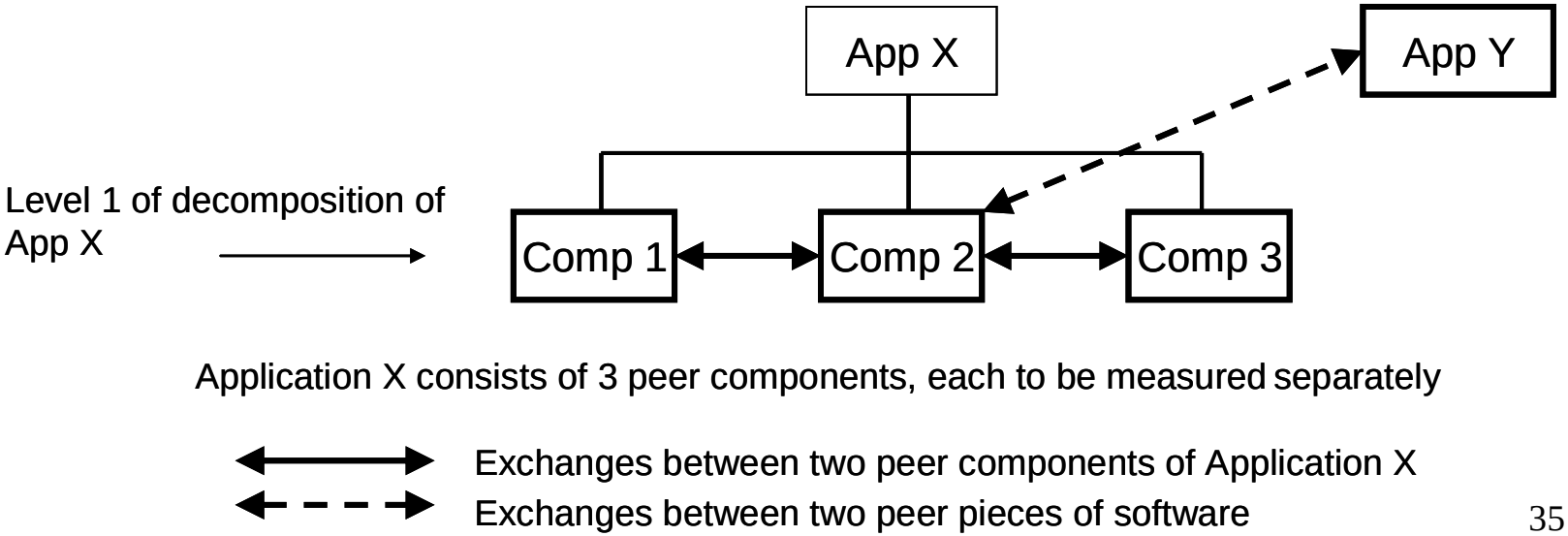
对等构件的定义

- 对等的定义
  - 处在同一层中的软件块彼此是对等的。
  - 注意：这2个对等的软件块未必在分解的同一个层次上
- 对等构件的定义
  - 将同一层中的软件在相同的分解层次上进行分解得到了一组互相合作的构件，每个构件完成了用户功能需求的一部分，这些构件即为对等构件。
  - （One component of a set of co-operating components, all at the same level of decomposition, that results from dividing up a piece of software within one layer, where each component fulfills a portion of the Functional User Requirements of that piece of software.）
  - 注意：软件块分解成的对等构件是为了响应功能和/或非功能性用户需求

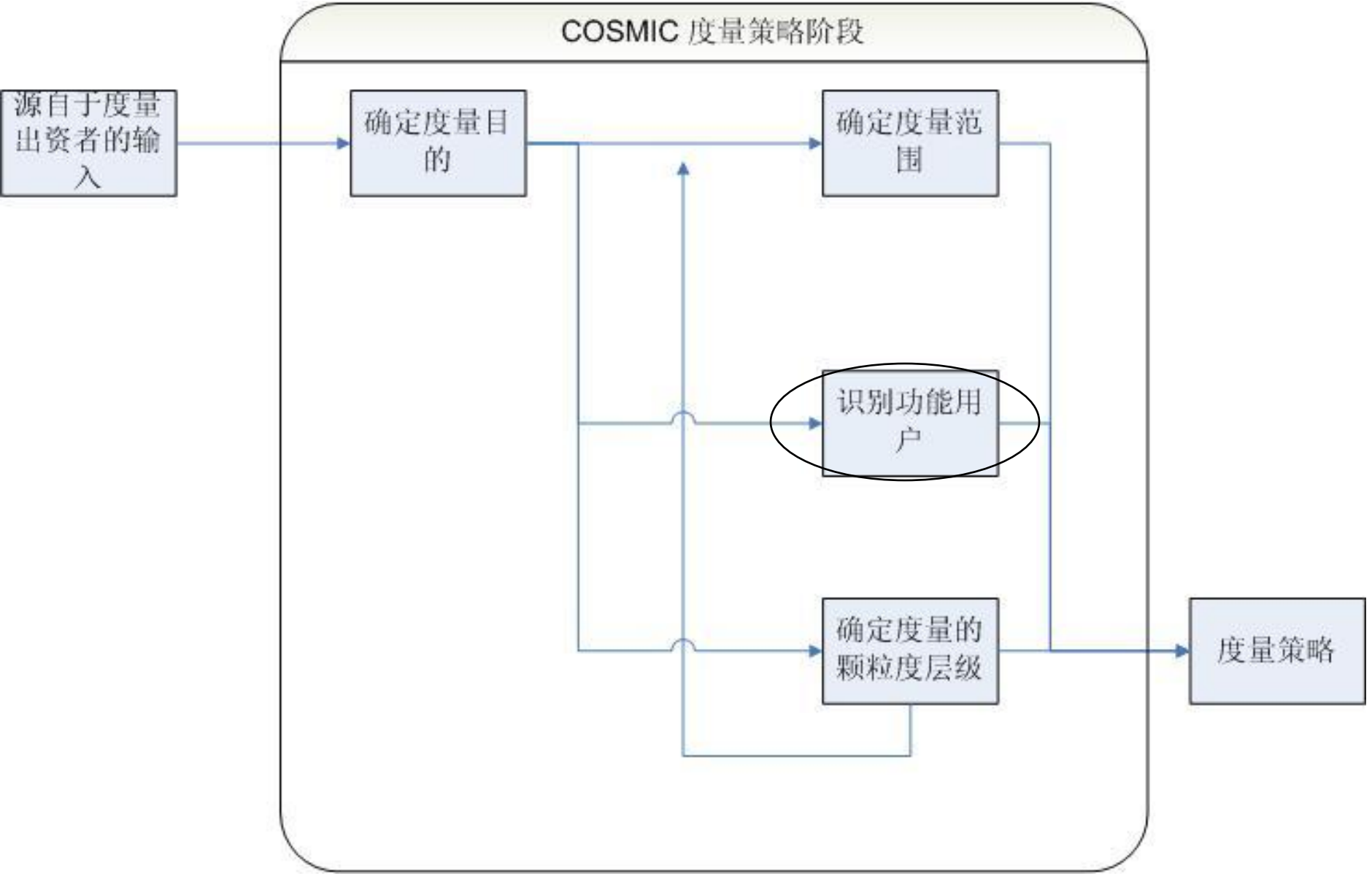
识别对等构件的原则

Measures

- a) 在同一层中的软件块的对等构件集合中，不同于层次之间，对等构件之间没有任何层次结构依赖。同一层中的所有的软件块的对等构件的用户功能需求都处在层次结构同一个层次。
- b) 软件块的所有对等构件必须相互协作以便软件块能够正确运行。
- c) 在两个对等构件之间可以通过数据组进行直接交换数据，一个对等构件的输出是另一个对等构件的输入，也可以是间接的交换数据，一个对等构件写数据到持久存储介质，而另一个对等构件随后读数据。



度量策略阶段-识别功能用户



识别功能用户

- 定义：用户功能需求中发送和/接收数据的一个(一类)用户
- 注意：
  - 功能用户可以是：
    - 人
    - 设备
    - 其他系统
  - 功能用户不同，规模不同
    - 比如，同一座房子对于建筑商、房东、保洁员、装饰公司面积是不同的
    - 比如，对于一部手机，人和设备都是功能用户，功能用户不同提供的功能不同，规模也是不同的。

识别功能用户的规则

- a）被度量的软件块的功能用户应该派生于度量目的
- b）当度量软件块的规模的目的与开发和维护软件块的工作量相关时，功能用户就应该是新的或者修改后的功能的功能用户

边界的定义

Measures

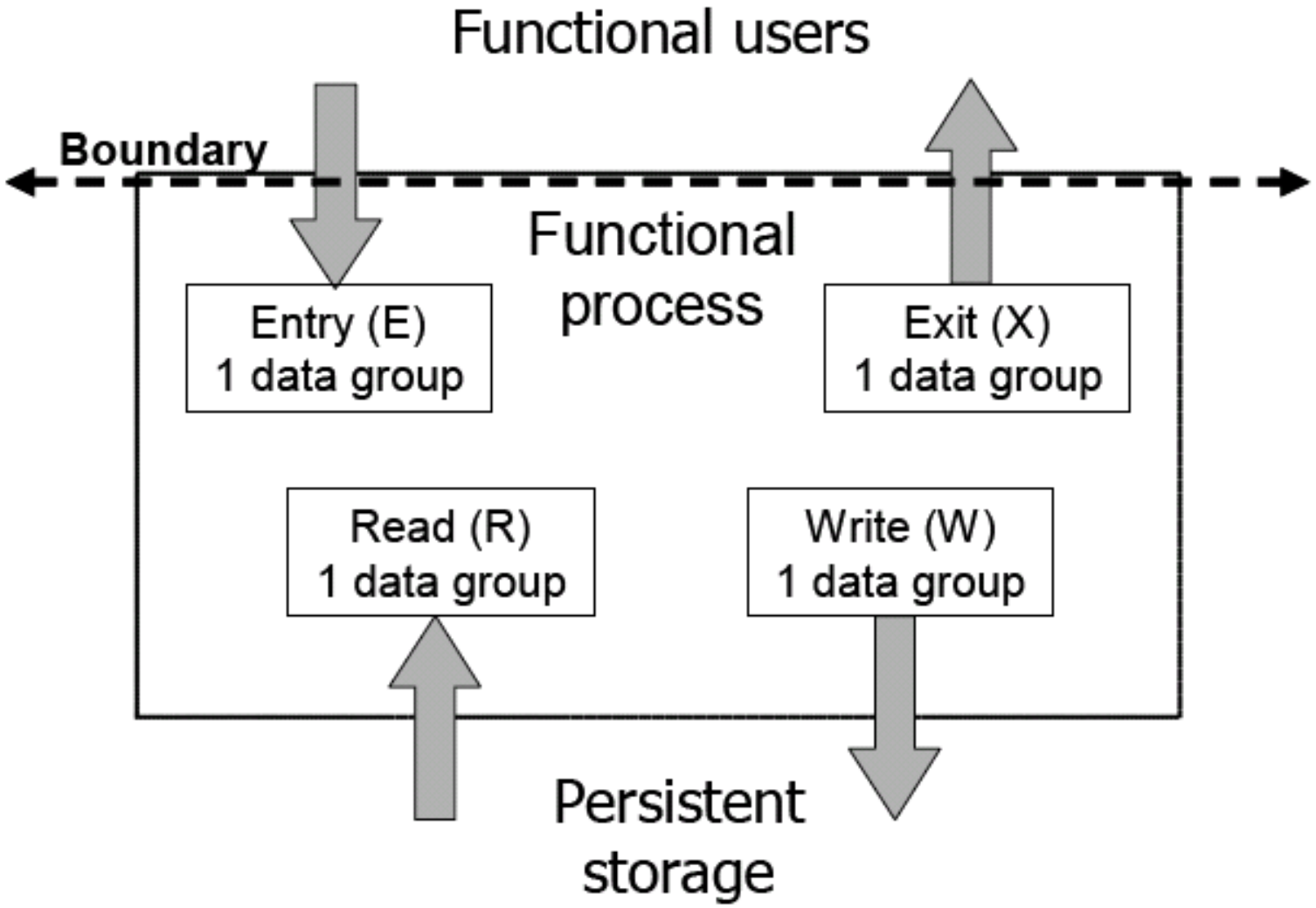
- 边界的定义
  - 边界定义为被研究的软件与其功能用户之间的概念性接口
- 注意：
  - 软件块的边界是其与它的运行环境之间的概念性的界线，是从外部用户的角度来考察． 边界使得度量人员能区分开什么包括在被度量的软件内部，边界允许度量者无二义性地将被度量的软件从其运行环境中区分开来．边界不用来定义度量范围。

识别边界的规则

Measures

- 识别与被度量软件交互的功能用户，边界位于功能用户和该软件之间。
- 在识别的一对层之间存在一个边界，一层是另一层的用户，另一层被度量。
- 在任意2个软件块之间存在一个边界，这2个软件块是对等，一个是另一个的功能用户。
- 注意：
  - 持久存储介质在边界内

边界示例

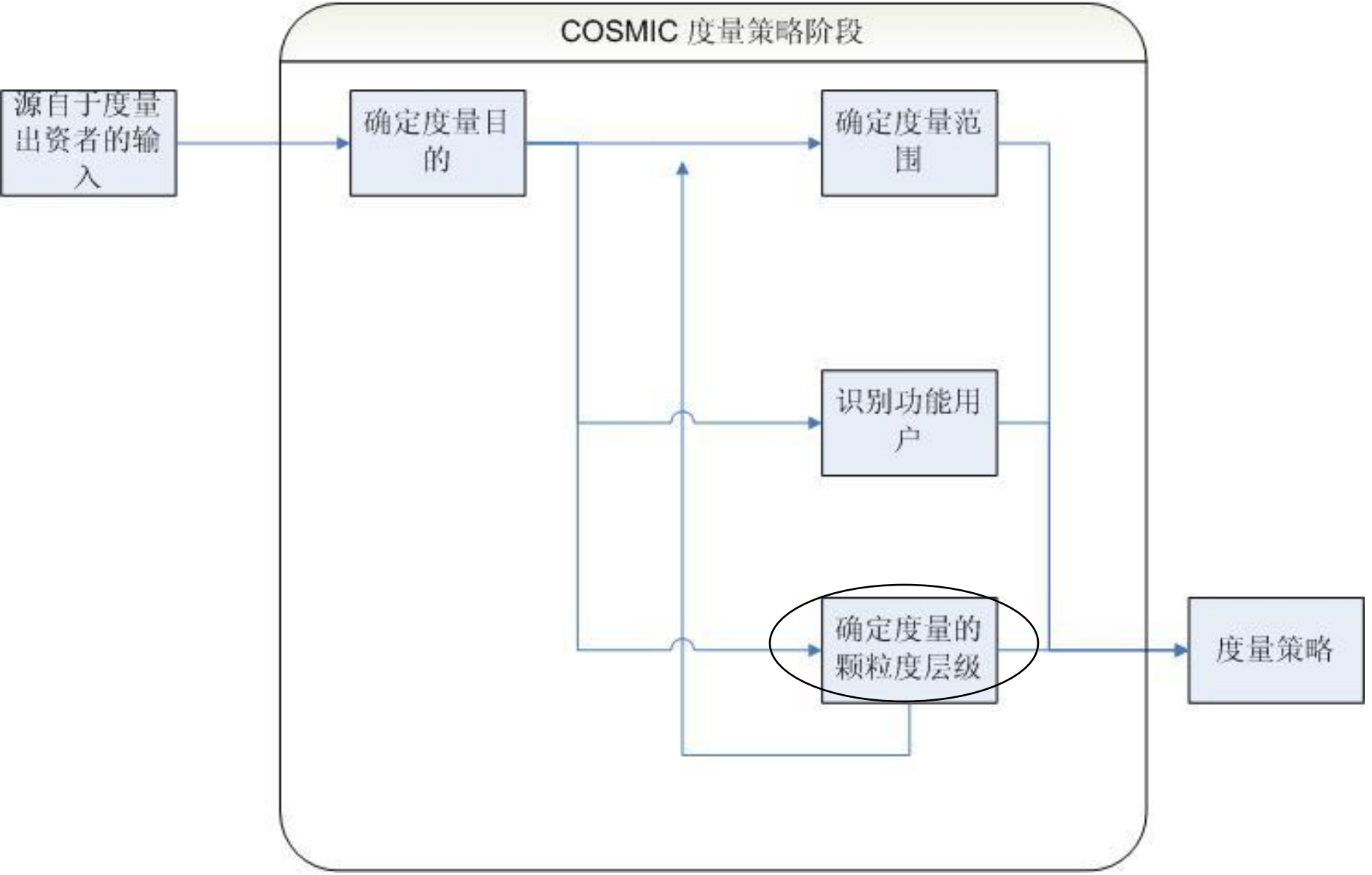

Functional users
Boundary
Functional process
Entry (E)
1 data group
Exit (X)
1 data group
Read (R)
1 data group
Write (W)
1 data group
Persistent storage

练习：确定度量策略

Measures

- 以某个人的手机为例，拟估算手机中应用软件的规模请：
  - 定义度量的目的
  - 定义度量的范围
    - 软件块
    - 限定在一层中
  - 识别功能用户
    - 人
    - 设备
    - 系统

度量策略阶段-确定度量的颗粒度层级



定义-颗粒度层级

Measures

- 在任何层次上的对单个软件块的扩展描述（例如，需求的描述、该软件块的结构描述）。每一次进一步的扩展，该软件块的功能描述都是在一个统一的、渐增的详细程度。
- 注意：
  - 度量人员应该意识到当需求在软件项目的早期进化的过程中，在任何时候，需要的软件功能的不同部分通常以不同的颗粒度级别被文档化。
- 类比：
  - 地图的缩放比例
    - 地图A只显示了高速公路和一级公路
    - 地图B显示了所有的高速公路、一级公路和二级公路
    - 地图C显示了所有的路并带有名称

定义-功能处理的颗粒度层级

Measures

- 一个描述软件块的颗粒度层次，在该层级上：
  - 功能用户是一个人或者是工程设备或者是一个软件块(并且不是这些的群体)
  - 发现软件块必须响应的单个事件的发生(而不是在定义事件组的任何一个级别)
- 注意：
  - 实践中，包含FUR的软件文档经常在不同的颗粒度层级来描述功能，特别是当文档仍在演变的过程中
  - “这些的群体”(功能用户)可能是，例如，一个“部门”，他们的成员处理很多类型的功能处理；或者是有很多种类型的仪器仪表的“控制面板”；或者是“中央系统”
  - 以比较高层的颗粒度层级描述的“事件组”，例如：通过会计软件系统的“销售事务”输入流，或者是“航空软件系统”的“导航命令”。

规则-功能处理的颗粒度层级

- a)应该在功能处理的颗粒度层级执行功能规模度量
- b)当某些FUR需要进行功能规模度量时，这些FUR还没有细化到所有的功能处理都可以被识别出来，并且所有的数据移动的细节还没有定义，应度量已定义的功能，然后对其按比例细化到功能处理的颗粒度层级

近似估算的处理

- COSMIC 建议将功能处理的颗粒度层级作为标准的颗粒度层级，所有的需求细化到该层级后再进行度量规模，否则就需要采用近似估算方法估出结果后再缩放到该层级。
- 颗粒度层次的缩放不影响度量的范围，但是由于展示的细节不同，对估算的结果是由影响的。
- 近似估算方法：
  - 可以度量每个USE CASE的平均功能点个数，通过估计USE CASE的个数估算功能点
  - 可以度量每个功能处理的平均功能点个数，通过估计功能处理的个数估算功能点

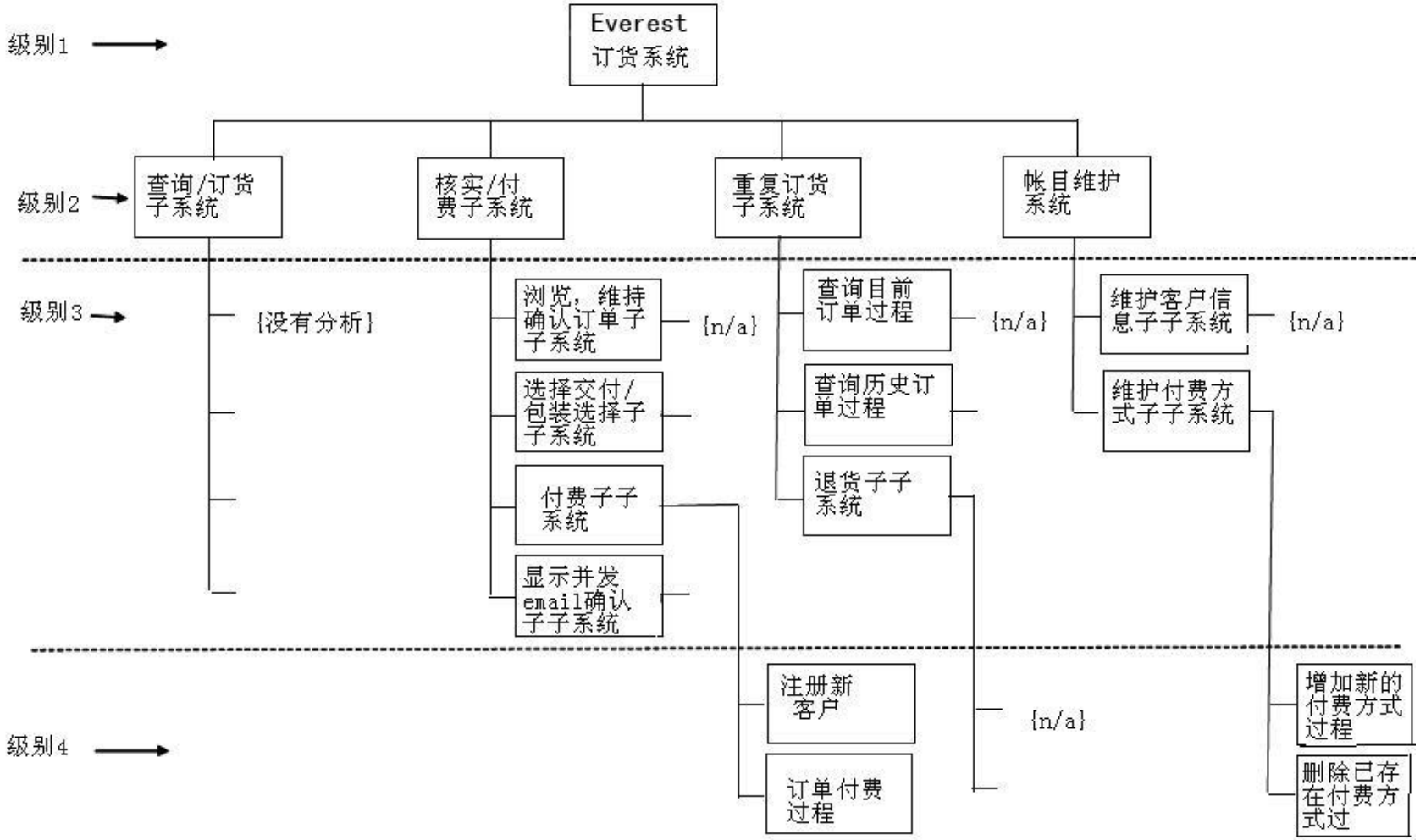
颗粒度层级的案例：因特网订货系统

Measures

- 为阐述粒度级别，下面的描述高度简化。描述仅包括 Everest 顾客用户可用的功能。因此，它没有包含系统为完成供货给客户而必需具备的功能，如 Everest职员、产品供应商、广告商、付费服务商等使用的功能。
- 度量范围：
  - 顾客通过因特网可访问的 Everest 应用系统部分。
- 度量目的：
  - 为了确定应用软件中顾客用户（功能用户）可用的功能规模。

因特网订货系统的颗粒度层级

Measures



Measures

映射阶段

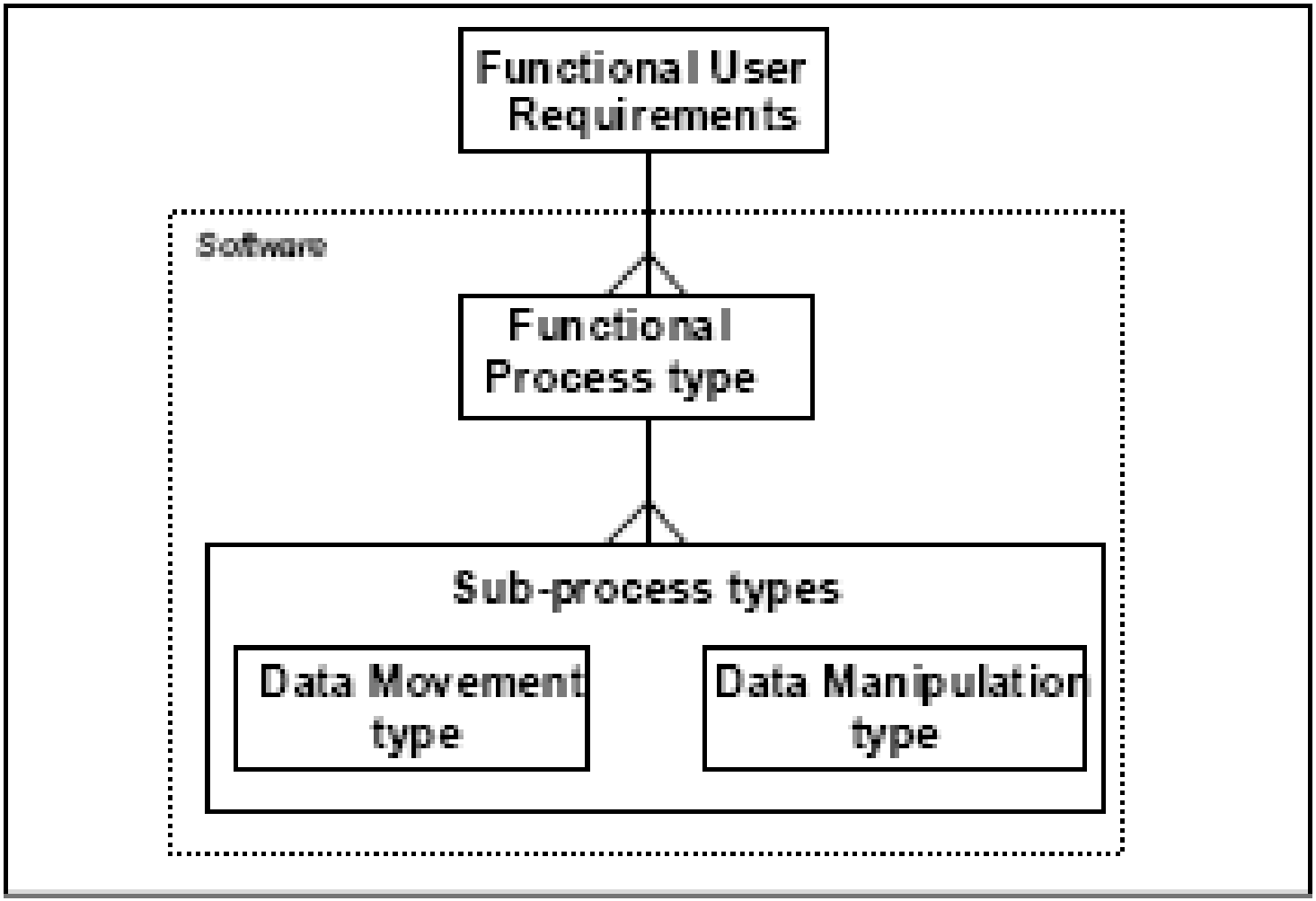
映射是将用户功能需求映射为通用软件模型

识别功能处理

识别数据分组

识别数据属性

通用软件模型


Functional User Requirements
Software
Functional Process type
Sub-process types
Data Movement type
Data Manipulation type

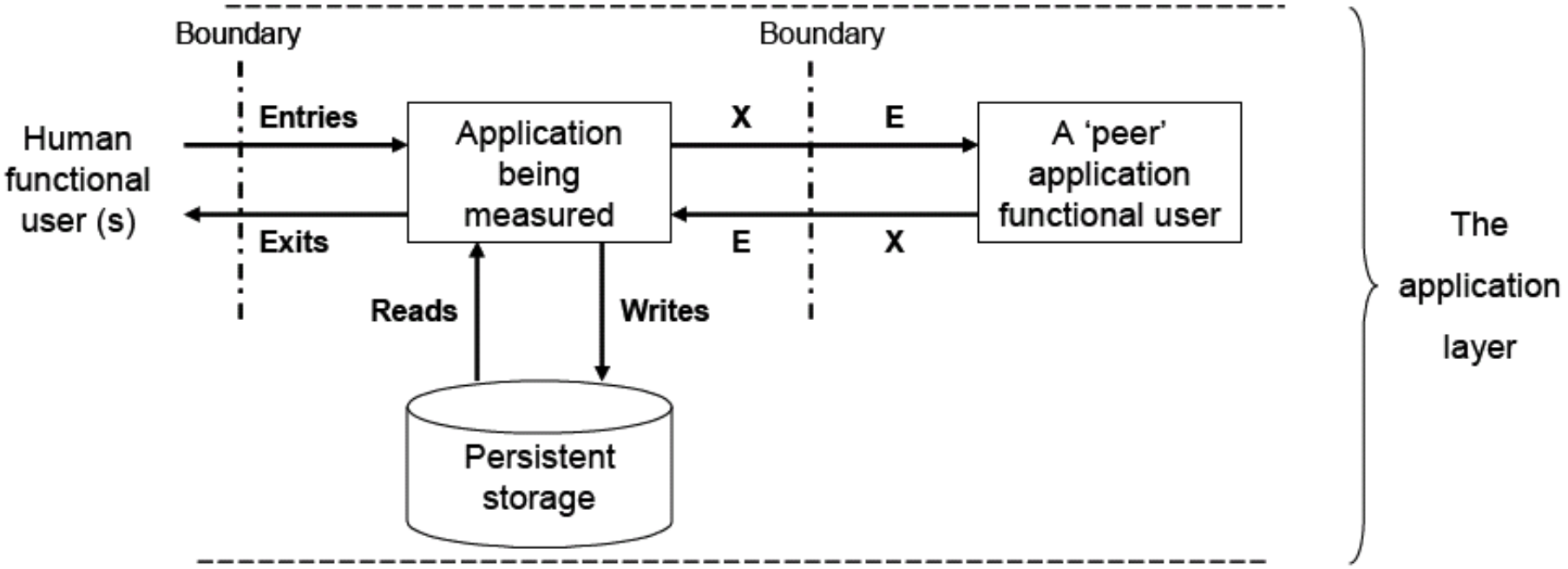
COSMIC 通用软件模型的原则

- a）软件从功能用户中接收到**输入**数据并为功能用户产生**输出**，和/或者其他的结果
- b）被度量的软件块的用户功能需求可以映射为唯一的一组功能处理
- c）每个功能处理由一系列**子处理**组成
- d）一个子处理可以是一个数据移动或者是一个**数据运算**
- e）每个功能处理均由来自于功能用户的输入数据移动所触发，该数据移动告知功能处理功能用户已识别了一个事件
- f）一个数据移动仅移动一个**数据组**
- g）一个数据组包含唯一的一组**数据属性**，这些数据属性描述了一个单一的**兴趣对象**
- h）数据移动有四种类型。一个**输入**从功能用户处向软件内部移动一个数据组。一个**输出**从软件内部向功能用户移动一个数据组。一个**写**从软件向持久存储介质移动一个数据组。一个**读**从持久存储介质向软件移动一个数据组
- i）一个功能处理应至少包含一个输入数据移动和一个写数据移动或者是一个输出数据移动，也就是说，一个功能处理应至少包含两个数据移动
- j）作为一种度量目的的近似，数据运算子处理不单独度量；我们假设，任何数据运算的功能都通过与其相关联的数据移动来计数。

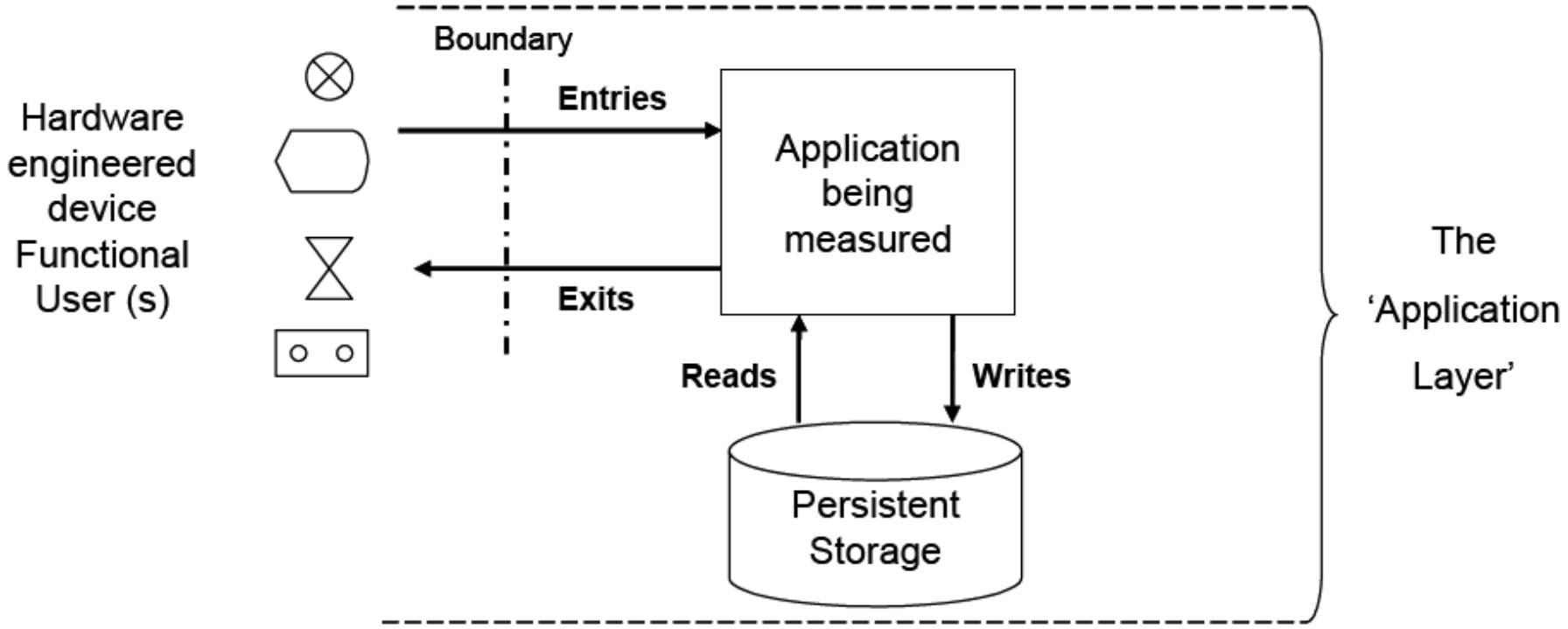
应用通用软件模型的原则

- COSMIC通用软件模型应该应用于每一个单独定义了度范围的单个软件部分的用户功能需求
- “应用COSMIC通用软件模型”意味着：
  - 识别一系列在用户功能需求中识别出来的，并且每一个功能用户(类型)都非常敏感的触发事件，
  - 识别为响应这些事件而必须相应提供的功能处理、兴趣对象、数据组和数据移动

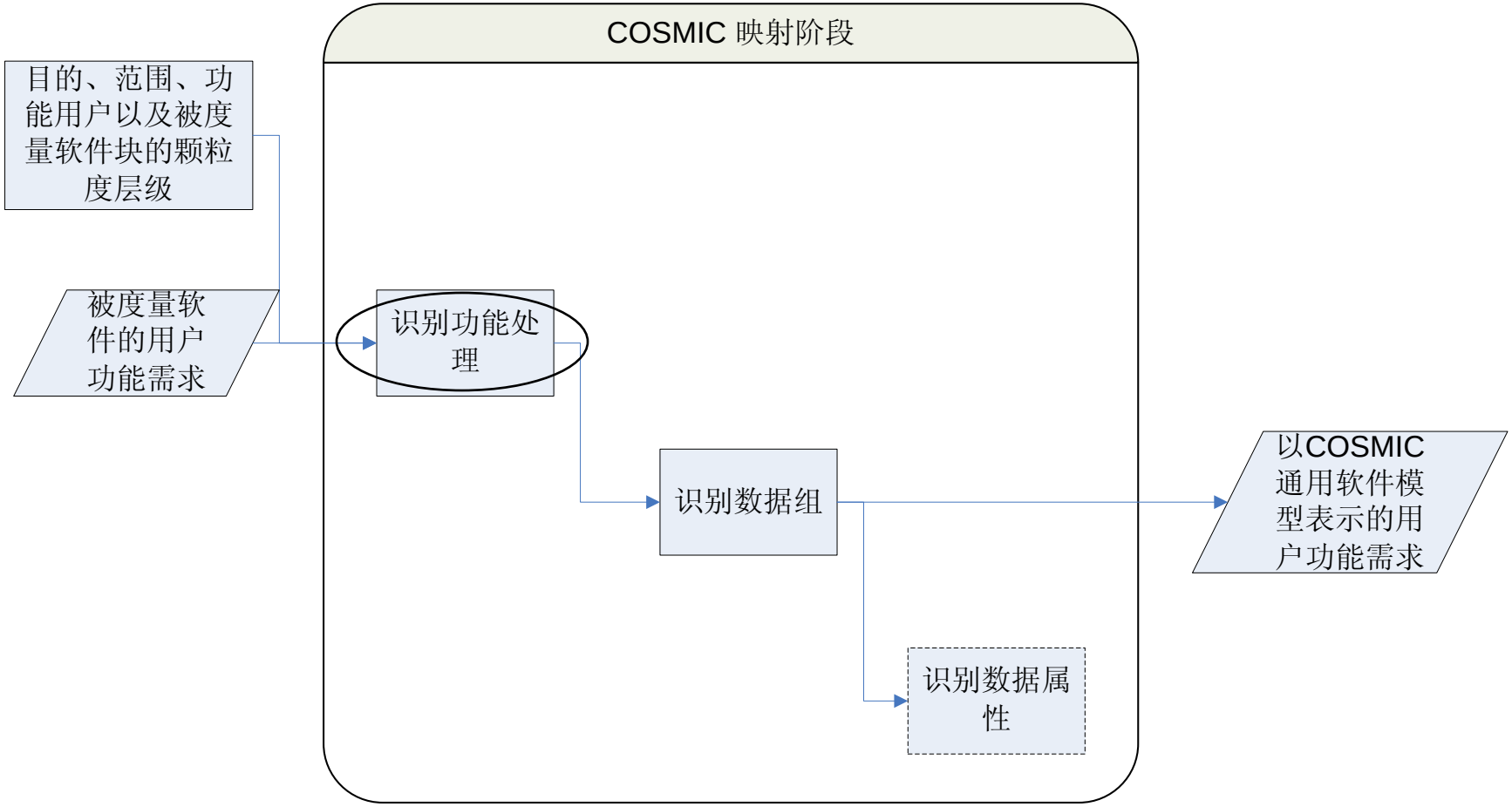
应用软件的通用软件模型示例



嵌入式软件的通用软件模型示例



映射阶段的过程模型


COSMIC 映射阶段
目的、范围、功能用户以及被度量软件块的颗粒度层级
被度量软件的用户功能需求
识别功能处理
识别数据组
识别数据属性
以COSMIC通用软件模型表示的用户功能需求

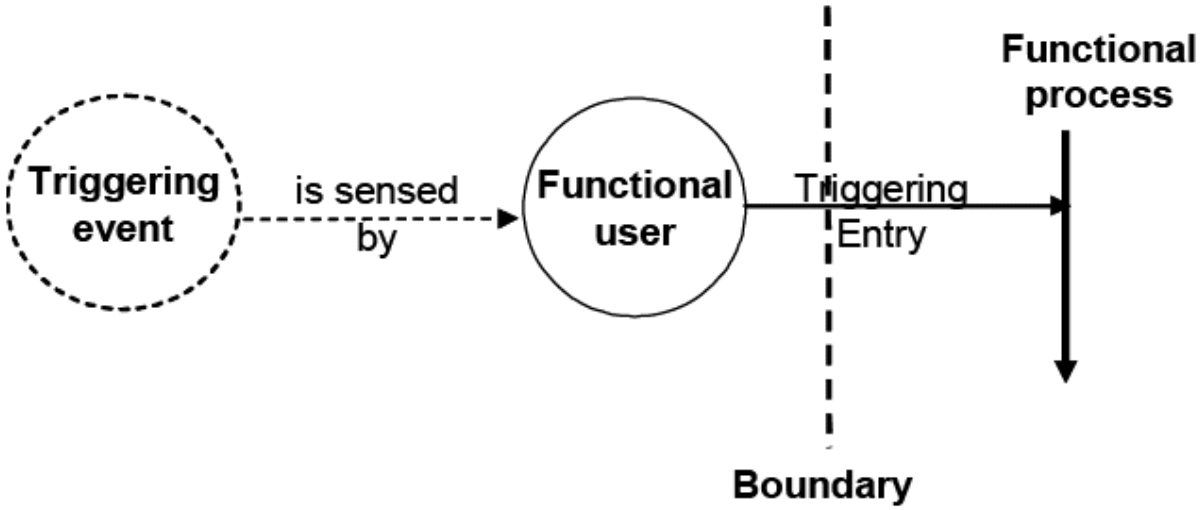
功能处理的定义

Measures

- 一个功能处理是用户功能需求集合的一个基本部件，包括一组唯一的，内聚的，可独立执行的数据移动.
- 功能处理由一个来自于功能用户的数据移动（一个输入）触发，该数据移动告知软件块功能用户已识别了一个触发事件。当它执行完所有它被要求的任务之后，功能处理结束。
- NOTE：
  - 除了通知软件块事件已经发生了之外，触发“输入”还可能包含与事件相关的兴趣对象的数据

触发事件的定义

- 导致软件块的功能用户启动（触发）一个或多个功能处理的事件. 在用户功能需求集中，导致一个功能用户触发一个功能处理的每一个事件：
  - 在那个用户功能需求集中不能再被细分，并且
  - 要么发生要么不发生
- Note:
  - 时钟和定时事件可以是触发事件.



识别功能处理的规则-1/2

Measures

- a) 一个功能处理至少从已认可的度量范围内的一个可识别出来的FUR中派生出来.(可反向追踪到功能需求)
- b)当一个可识别的触发事件（类型）发生时,一个功能处理（类型）被执行.(可关联到某个触发事件,必须由外部触发)
- c)一个特定的事件(类型)可能会触发一个或多个并行执行的功能处理(类型)。一个特定的功能处理(类型)可能由不止一个事件(类型)触发。 (触发事件与功能处理是多对多的关系)
- d)一个功能处理至少有2个数据移动组成,一个输入加上一个输出或者写.
- e) 一个功能处理属于且仅属于某一层的度量范围内的软件块.
- f) 在实时软件环境中，当进入自感应的等待状态时可以认为功能处理就结束了（例如，为响应触发事件，功能处理已经完成了其所有必须执行的一切，并直到其收到下一个触发输入）。

识别功能处理的规则-2/2

Measures

- g）即使一个功能处理伴随着输入数据最多属性的不同的子集，也应该识别为一个功能处理(类型)，并且，即使这种差异和/或不同的输入数据值可能引发功能处理的不同处理路径。
- h)单独的事件(类型)和与之相关的单独的功能处理(类型)在如下情形下应该区别对待：
  - 当决策导致了在时间上不连续、拆分的事件时(例如，今天输入订单数据，而后确认订单是否收到，这都需要单独的判断，这些都应该被看作是单独的功能处理)
  - 当活动的职责是分立的(例如，在一个人事系统中维护基础个人信息的职责与维护职工工资数据就是单独的功能处理；或对于一个已上线的软件包，系统管理员使用的维护软件包参数的功能应区别于“常规”功能用户使用的功能)

在商业应用软件中识别功能处理-1

Measures

- 案例1：人触发功能处理
  - 在一个公司里，收到一个订单(触发事件)，这就需要一个员工(功能用户)去输入订单数据(触发输入与兴趣对象“订单”相关的数据），作为订单输入这个功能处理的第一个数据移动
- 案例2：软件触发功能处理
  - 假设上述例子中是接收订单，订单应用程序必须将客户信息发送给要被度量的客户订购中心应用程序。现在，订单应用程序已经成为中心应用程序的一个功能用户了。订单应用程序感知接收到客户数据的事件，并触发了客户登记中心应用程序存储这些数据的功能处理，作为一个触发输入，通过发送“客户”兴趣对象数据给中心应用程序。

在商业应用软件中识别功能处理-2

Measures

- 案例3：时钟信号触发
  - 假设用户功能需求是报告该年度最后的业务产出的批处理，并且重新设置下一年起始点。按照自然规律，操作系统产生的年底“时钟报时信号”导致包含一个或多个功能处理的批处理流启动一个事件可能触发一个或多个单独执行的功能处理。
- 案例4：一个触发事件触发多个功能处理
  - 周末“时钟信号”启动报表的生成，以及启动工作流系统中评审有效期的过程。
- 案例5：一个功能处理由多类触发事件触发
  - 在银行系统中，对账单可由月底的批处理过程触发，也可能由客户的特定的请求触发。

实时软件示例

- 案例1：触发事件通常由传感器检测到
  - 当温度达到一个特定的值(触发事件)时，传感器(功能用户)给报警灯(功能处理)开关发送信号(触发输入数据移动)
- 案例2：定期的时钟信号也会触发功能处理
  - 在某些实时过程控制软件中，时钟(功能用户)的一次滴答声引发整个时钟发送一个信号(触发输入)，一个单信息消息使功能处理重复其正常控制周期。功能处理继而读取不同的传感器，接收兴趣对象有关的数据，执行必需的行为。没有其他伴随时钟报时信号的数据了。
- 案例3：一个事件可能会触发一个或者多个单独且并行执行的功能处理。
  - 在核电厂中探测到的事故状态可能会触发相互独立的功能处理，这些功能处理来自于核电厂内不同部分，以降低控制棒、启动制冷引擎、关闭阀门、拉响警报装置等。
- 案例4：一个功能处理可能由不止一种类型的触发事件触发
  - 飞机收起轮子可能由传感器或者是飞行员的命令触发

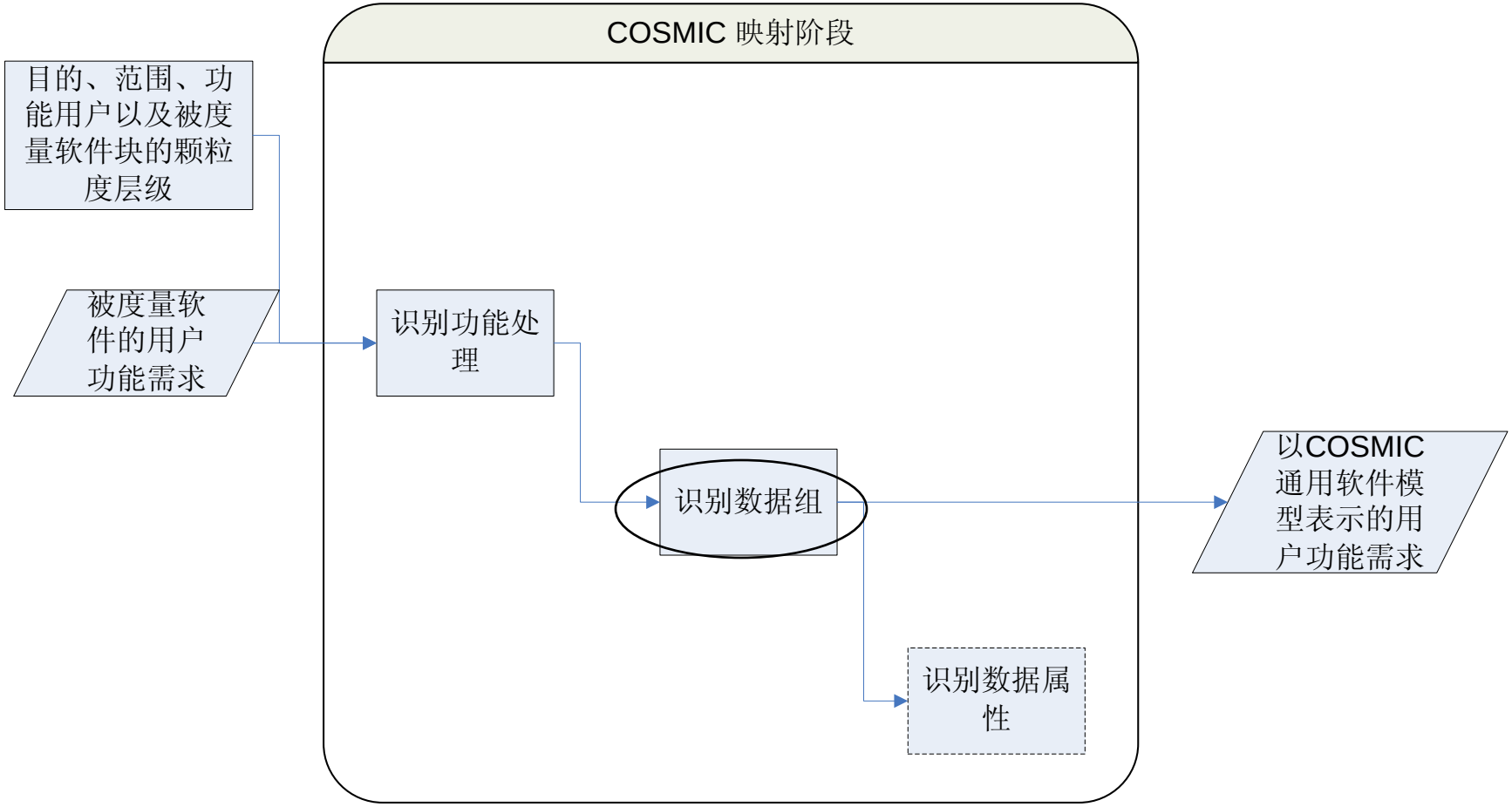
练习：识别功能处理

- 假定有一个最简单的手机，只具有如下的功能：
  - 打电话
    - 可以直接输入电话号码拨出电话
    - 可以从通话记录中查找电话号码拨出电话
    - 可以从通讯录中查找电话号码拨出号码
  - 接电话
    - 可以接听电话
    - 可以拒绝接听电话
    - 可以不接电话
    - 占线时无法接听其他打入的电话
  - 维护通讯录（人名、电话）
    - 增加
    - 修改
    - 删除
    - 查找某个人
- 请识别出触发事件、功能用户及功能处理

参考答案：

- 拨出电话
- 查找联系人
- 查找通话记录
- 接收到电话
- 接听电话
- 增加联系人
- 删除联系人
- 修改联系人

映射阶段的过程模型


COSMIC 映射阶段
目的、范围、功能用户以及被度量软件块的颗粒度层级
被度量软件的用户功能需求
识别功能处理
识别数据组
识别数据属性
以COSMIC通用软件模型表示的用户功能需求

兴趣对象的定义

Measures

- 从用户功能需求角度识别的任何事物。它可能是任何物理的事物，也可能是任何功能用户世界中的概念对象或者概念对象的一部分，这些都是软件必须处理和存储的数据。
- 注释：在COSMIC方法中，使用术语“兴趣对象”以避免与特定的软件工程方法相关的术语。这个术语并不暗示这个“对象”就是面向对象方法中的对象

数据组的定义

Measures

- 一个数据组是一个有区别的，非空的，没有顺序的并且没有冗余的数据属性的集合，包含的每个数据属性描述了同一个兴趣对象的一个互补的侧面。
- 注意：
  - 数据组是兴趣对象的属性的子集
  - 一个兴趣对象可能包含多个不同的数据组
  - 不同的数据组具有不同的属性
  - 如果有2组具体的数据，属性相同，数据的行数不同也是同一个数据组

识别数据组的原则

Measures

- a）通过它的唯一的数据属性集合，每个被识别的数据组必须是唯一的和可区别的.（不重复）
- b）每个数据组必须被直接的关联到一个在软件的FUR中描述的兴趣对象上.（用户可见）
- c）一个数据组必须在支持这个软件的计算机系统中被具体化.（有物理的载体）

持久存储介质的定义

Measures

- 持久存储介质是在能让功能处理结束后存储数据组的存储介质，并/或一个功能处理可以从它检索其他功能处理存储的或同一个功能处理在更早的时间存储的、或者某些“其他处理”存储的数据组
- 注释1：在COSMIC模型中，因为持久存储介质是在软件边界内，所以，就不能将其视为软件的一个用户
- 注释2：“其他处理”的一个例子：只读存储器的制造

数据组的4种具体化的表现形式

Measures

- 在实践中,数据组的具体化表现为4种形式:
  - -a)通过一个物理的记录结构在一个持久的存储设备上(如文件,数据表,ROM等).
  - -b) 通过计算机中的易失性存储器的物理结构(动态分配的数据结构或通过预先分配的内存空间块).
  - -c) 通过成组的功能有关的数据属性的展示在I/O设备上 (显示屏幕, 打印的报告, 控制面板显示器, etc.)
  - -d)在设备和计算机之间或网络上传输的消息

识别数据组的案例-商业应用软件.

Measures

- 用户功能需求：
  - 我对一个人员数据库执行一个复杂查询来提取所有超过35岁的员工的姓名清单
- 识别的数据组：
  - 查询条件
  - 人员数据库
  - 姓名清单

识别数据组的案例-实时软件

Measures

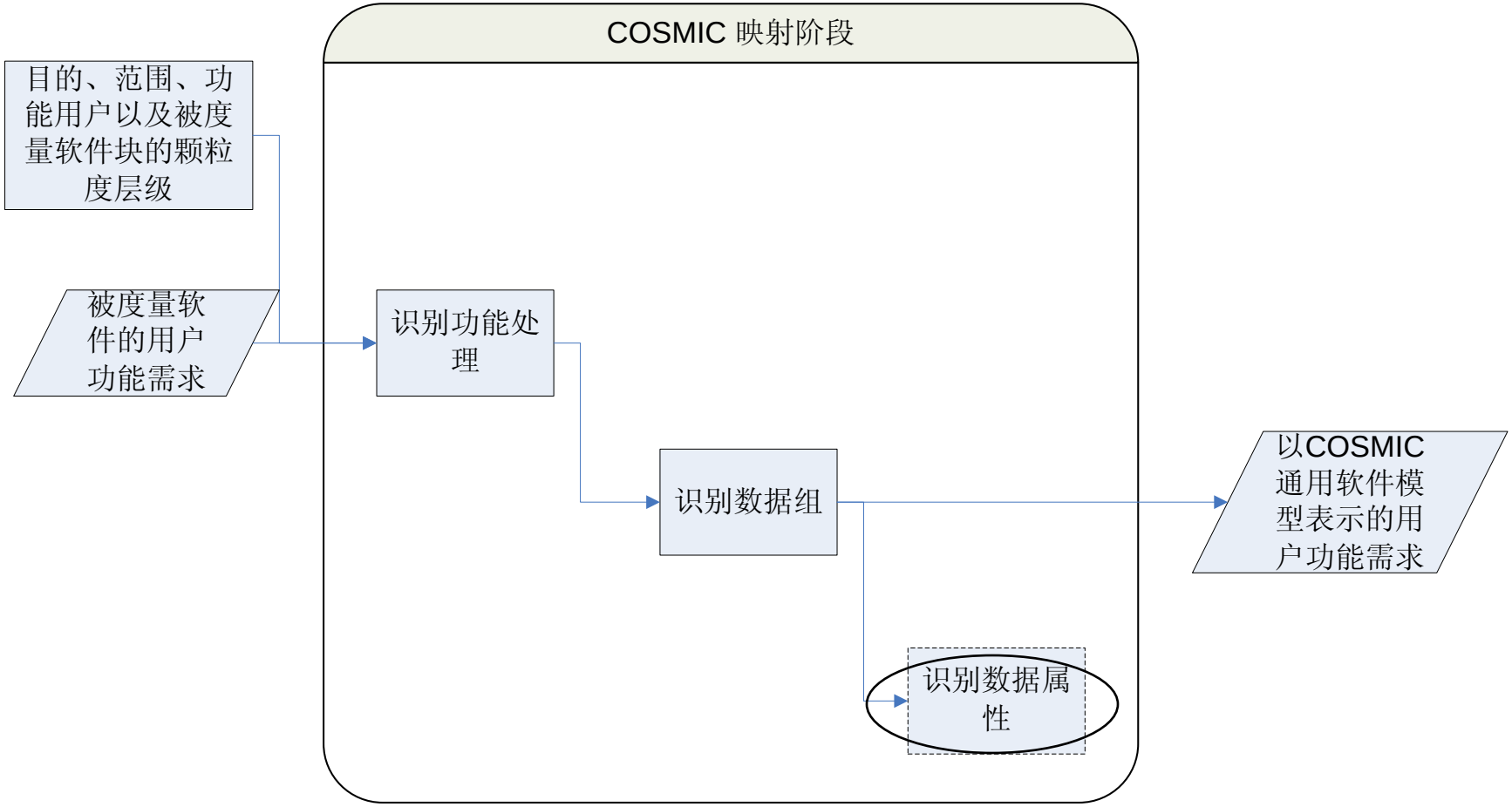
- 一个阀门是打开还是关闭的
- 一个信息交换(message switch)会接受一个信息数据组作为一个输入并作为一个输出无修改地转发它. 信息数据组的属性可以是进路电码(route code),并且兴趣对象是”信息”.
- 共用数据结构,表示了在FUR中提到的兴趣对象,保存在易失性存储器中并可以被被度量软件的大多数功能处理存取.
- 引用数据结构, 表示了在FUR中发现的图形或者表格,保存在持久性的存储器(如ROM)中并可以被被度量软件的大多数功能存取.
- 文件,通常被指定为”脱机文件”,表示了在FUR中提到的兴趣对象,保存在持久存储设备中.

不作为候选数据组的数据

Measures

- 任何出现在输入或输出画面或报告中的数据，并且这些数据与功能用户的兴趣对象无关的，那么这些数据就不应该被识别为数据组，所以，也不应该被度量。
  - 通常这类数据是显示在所有的画面中的页眉和页脚(功能名称、应用名称、系统日期等)
  - 确保功能用户控制软件使用的控制命令(仅仅在业务应用领域中定义的概念)，而不是进行数据移动，比如上下翻页命令、单击“OK”以表示收到错误信息等

映射阶段的过程模型


COSMIC 映射阶段
目的、范围、功能用户以及被度量软件块的颗粒度层级
被度量软件的用户功能需求
识别功能处理
识别数据组
识别数据属性
以COSMIC通用软件模型表示的用户功能需求

数据属性的定义

Measures

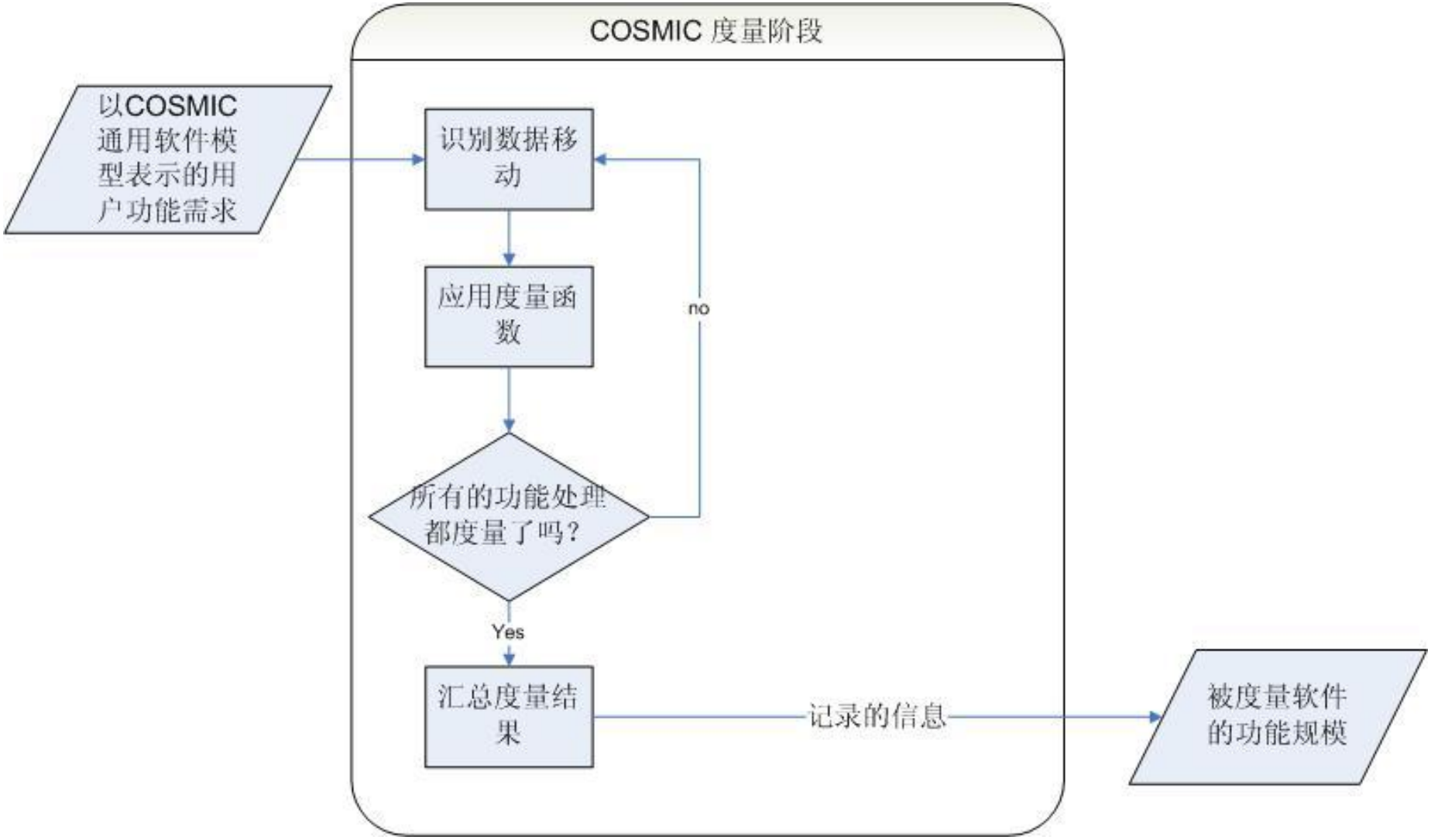
- 在一个已识别的数据组中，一个数据属性是最小的信息包，从软件的FUR的角度包含了一种含义.
- 例子
  - a）在商业应用软件中
    - 数据字典中的数据元素类型
    - 出现在概要或逻辑数据模型中的数据属性
  - b）在实时应用软件中
    - 从一个传感器接收到电压信号

Measures

度量阶段

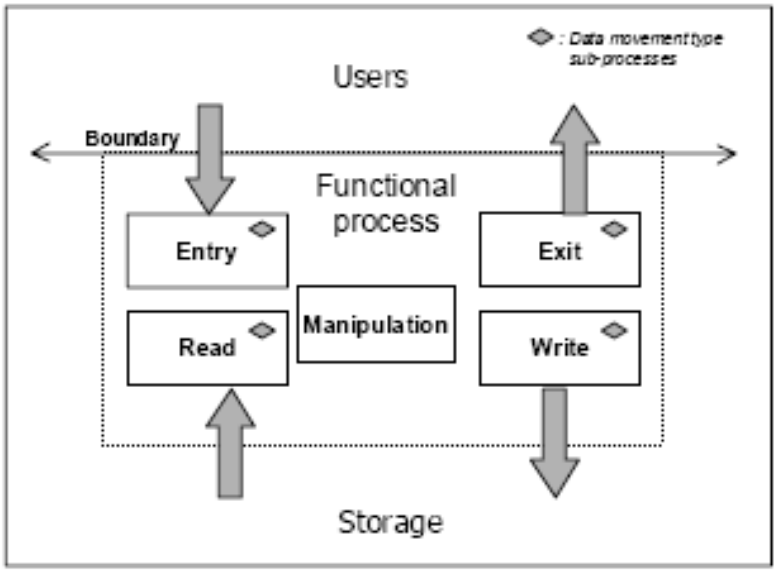
识别数据移动
应用度量函数
合计度量结果

度量阶段的过程模型


COSMIC 度量阶段
以COSMIC通用软件模型表示的用户功能需求
识别数据移动
应用度量函数
所有的功能处理都度量了吗?
no
Yes
汇总度量结果
记录的信息
被度量软件的功能规模

识别数据移动

- 定义:数据移动
  - 移动一个单独的数据组类型的功能构件
  - Note 1:有四种子类型的数据移动:输入,输出,读和写
  - Note 2:基于度量目的，每种数据移动子类型包括了特定的数据运算.
  - Note 3:更精确的讲，它是数据移动的发生，不是数据移动类型，它实际上移动的是数据组的实例（不是类型)。这也适用于输入、输出、读和写的定义。



输入

Measures

- 定义
  - 输入是一种数据移动，它从功能用户移动一个数据组通过边界到需要它的功能处理。
    - 注释：输入被认为包含了与之相关的某些数据运算
- 原则
  - 输入从功能用户处移动一个描述单个兴趣对象的数据组，穿越边界，移入功能处理。如果功能处理的输入由不止一个数据组组成，需要为输入的每一个唯一的数据组识别一个输入。
  - 输入不应该穿越边界输出数据，或者读或写数据
  - 输入包括了任意的“输入请求”的功能，除非功能用户需要被明确告知输入什么数据，在这种情况下，获取每一个数据组都需要一个输出/输入对

输入

Measures

- 规则
  - 触发输入的数据组可能仅包含一个数据属性，这个触发输入只是通知软件发生了事件Y。在商业应用软件通常是触发输入的数据组拥有多个数据属性，触发输入通知软件“事件Y发生了，这是与该特定事件相关的数据”
  - 时钟节拍作为触发事件经常认为应该在被度量软件之外。因此，例如，每3秒发生一次的时钟事件应该与输入仅有一个数据属性的数据组相关。需要注意的是，无论是由硬件周期产生的触发事件还是由被度量软件边界之外的另外一个软件块产生的触发事件都是没有任何区别的。
  - 除非需要一个特定的功能处理，否则从系统的时钟获取时间是不可以被认为是一个输入
  - 如果一个特定事件的发生触发了数据组输入，这些数据组是“n”个数据属性构成的特定兴趣对象，并且用户功能需求允许相同的事件可以触发该兴趣对象包含其属性子集的其他数据组的发生，则只识别1个由“n”属性构成的输入。

订单输入的案例-1

Measures

- FUR如下：
  - 输入包含多个产品的订单，订单的属性有：
    - 订单头(订单号,客户号,交付日期)
    - 订单项(订单号,项序号,产品号,数量)
  - 订单号和项序号有系统自动生成
  - 客户号和产品号必须验证正确性

订单输入的案例-2

Measures

- 功能点为7个:
  - -E 订单头
  - -R 客户
  - -E 订单项
  - -R 产品
  - -W 订单头
  - -W 订单项
  - -X 出错信息或确认信息

输出

Measures

- 定义
  - 输出是一种数据移动，它移动一个数据组从一个功能处理通过边界到需要它的用户。
    - 注释：输出被认为包含了与之相关的某些数据运算
- 原则
  - 输出是移动一个数据组从一个功能处理通过边界到它所要求的用户。如果功能处理的输出包含不止一个数据组，就将该输出中的每一个唯一的数据组识别为一个输出
  - 输出既不应该穿越边界输入数据，也不读或写数据
- 规则
  - 所有的由软件产生和输出的不包含用户数据(例如，错误消息)消息都应该被认为是一个兴趣对象的一个属性的多个值(例如，可能被命名为“错误提示”)。因此，当用户功能需求中明确要求的时候，在一个功能处理中仅识别1个输出以代表所有这些消息的发生。
  - 如果一个特定事件的发生触发了数据组输出，这些数据组是“n”个数据属性构成的特定兴趣对象，并且用户功能需求允许相同的事件可以触发该兴趣对象包含其属性子集的其他数据组的发生，则只识别1个由“n”属性构成的输出。

案例

Measures

- 在屏幕上输出：“存盘失败”，“年龄超出上限”等都不被识别为输出，不计算功能点
- 操作系统的提示不被识别为输出，不计算功能点，如：“网络连接失败”等
- 如果在提示信息中包含了用户感兴趣的数据，则需要识别为输出，如：
  - “你输入的金额大于了资信上限100万”
  - 如果在实时系统中提示“第3号传感器发生了故障”

案例：

- 1个数据组输出在屏幕上，并以相同的格式打印出来，识别为1个输出
- 1个数据组输出在屏幕上，但是以显著不同的格式打印出来，识别为2个输出
- 1个数据组以图形的方式输出在屏幕上，以数据表格的形式打印出来，识别为2个输出

读

Measures

- 定义
  - 读是一种数据移动，它移动数据组从持久存储到需要它的功能处理.
    - 注释：读被认为包含了与之相关的某些数据运算
- 原则
  - 读是从持久存储介质向功能处理移动一个描述单一兴趣对象的数据组。如果功能处理必须从持久存储介质取得不止一个数据组，则应该为检索的每一个数据组识别一个读
  - 读不应该穿越边界接收或输出数据或写数据
  - 在功能处理的过程中，计算或移动常量或者功能处理内部的并且只能由程序员来更改的变量、或者是计算过程的中间结果、或由执行功能处理而产生并存储的、而非来自于用户功能需求的数据，这些都不应该被看作为读数据移动
  - 读数据移动经常包括一些“读请求”功能(所以“读请求”功能不能被识别为单独的数据移动)。

写

Measures

- 定义
  - 写是一种数据移动，它移动存在于功能处理中的数据组到持久存储介质中。
    - 注释：写被认为包含了与之相关的某些数据运算
- 原则
  - 写是从功能处理向持久存储移动描述单个兴趣对象的单独的数据组。如果功能处理必须向持久存储移动一个以上的数据组，则将向持久存储移动的每个单独的数据组识别为一个写
  - 写不应该穿越边界接收或输出数据或读数据
  - 从持久存储删除数据组的需求应该作为一个单独的写数据移动进行度量
  - 在功能处理的过程中，当功能处理结束时数据的移动或操作并不持续下去、或者功能处理内部更新变量或产生计算的中间结果时，都不可以看作为一个写数据移动

数据移动的唯一性及例外规则

Measures

- a)除非FUR明确说明了，否则输入描述了一个兴趣对象的所有数据属性和相关的数据运算到一个功能处理中应被识别为1个输入（注意：一个功能处理当然可以处理多个输入，每次数据组的移动描述了不同的兴趣对象）。该规则也适用于输出、读、写。
- b）如果在FUR中明确描述了多个输入，在一个给定的功能处理中，这些输入移动了同一个兴趣对象的不同的数据组，则需要识别和计算多个输入数据移动（类型）。
- 类似地，如果在FUR中明确描述了多个输入，这些输入在同一个功能处理中移动了相同的数据组，但是每个输入具有不同的相关的数据运算类型，则需要识别和计算多个输入数据移动。
- 当在同一个功能处理中，多个输入来自于不同的功能用户，他们输入不同的数据组(每个数据组描述的是同一个兴趣对象），此时，就产生了如上的需求。对于输出、读、写以此类推。
- c)在任何一个功能处理中，重复发生的数据移动类型(例如，采用相同数据运算移动相同的数据组)不应该被识别和计算超过1次
- d)如果给定的功能处理中一个数据移动类型发生多次，多次发生的数据移动具有不同的数据运算，被移动的数据组的不同的数据属性值产生了不同的处理路径，则不可以将数据移动类型识别和计算多次。

- （数据移动　　兴趣对象　　数据组　　数据运算）

应用度量函数

- COSMIC-FFP度量函数是一个数学函数,参数是数据移动类型,返回值是FFP定义的值,计量单位是CFP（Cosmic Function point）,用以度量数据移动的规模.

汇总度量结果的规则

Measures

- 可加规则：
  - a)对每一个功能处理:Size（功能处理i）= size(输入i) + size(输出i) + size(读i) + size(写i)
  - b)对任何一个功能处理:Size(Change(功能处理i)) = size(增加的数据移动) + size(修改的数据移动)+ size(删除的数据移动)
  - c)度量范围内的软件块的规模可以根据规则e和规则f对其包含的功能处理的规模累计得到
  - d)度量范围内的软件块的变更规模可以根据规则e和规则f对其包含的功能处理的变更规模累计得到
  - e)如果FUR是在相同的功能处理颗粒度层次，同一层内的软件块的规模或软件块变更的规模可以累加在一起
  - f)如果对于度量目的有意义，在任何一层或不同层的软件块的规模和/或软件块变更的规模可以累加在一起
- 不可加规则：
  - g)软件块的规模不能通过累加其构件的规模而得到（忽略软件是如何分解的），除非消除了内部构件之间交互而产生的数据移动
  - h)如果COSMIC方法进行了本地扩展，则扩展后的方法得到的规模不能和标准COSMIC方法得到的规模进行累加

Measures

简单案例

客户信息维护的案例-1

Measures

• 增加一个新客户			4
– 用户输入		ENTRY	1
– 测试名字是否存在		READ	1
– 显示错误信息		EXIT	1
– 存储客户数据		WRITE	1
• 修改客户数据			6
– 用户输入		ENTRY	1
– 取出客户数据	READ		1
– 显示错误信息	EXIT		1
– 显示客户数据		EXIT	1
– 输入修改的数据		Entry	1
– 存储修改数据		WRITE	1

客户信息维护的案例-2

Measures

- 删除客户 4
  - -用户输入 ENTRY 1
  - -读取客户数据 READ 1
  - -显示错误信息 EXIT 1
  - -删除客户 WRITE 1
- 接收付款 4
  - -用户输入 ENTRY 1
  - -读取客户数据 READ 1
  - -显示错误信息 EXIT 1
  - -存储数据 WRITE 1

订单处理的案例-1

Measures

- 案例1:在订单处理软件系统中存储了关于客户的一些信息,如:
  - -customer_ID
  - -customer_name
  - -customer_address
  - -customer_contact
  - -Telephone
  - -customer_credit_limit
  - -customer_type_code
  - -date_of_last_order

订单处理的案例-2

Measures

- 这些信息必须被输入进去.显然,‘Customer’是一个OOI. 对于最简单的输入客户信息的功能处理而言,包含了1个是输入和一个写的数据移动,而且还可能包括了一个输出错误信息的数据移动,因此可以识别为:3 CFP.
- 案例2:注意在上面的属性中, ‘customer_type_code’包含了多个代码值,如 P, R, W等用来代表个人(Personal),零售商(Retailer), 批发商(Wholesaler).
- 假定这个系统还存储关于客户类型的其他数据,如:
  - -cust_type_description
  - -cust_type_order_value_discount_%
  - -cust_type_payment_terms

  等等.

订单处理的案例-3

Measures

- 此时客户类型就成为了一个OOI，当输入客户信息时，需要读取客户类型的信息以验证customer_type_code是否正确. 但是并没有识别一个单独的输入. 当customer_type也是OOI时，输入客户信息的功能处理增加了1个读数据移动，因此总的规模是: 4 CFP.
- 案例3:假定 customer_type ‘thing’只有属性 ‘customer_type_code’和 ‘customer_type_description’. 系统提供多种类型的‘customer_type_description’供用户选择输入，此时不能认为customer_type 是单独的OOI，因此识别建立客户信息的功能处理时，只能识别为是3 CFP.
- 案例4:假定属性‘customer_type_code’and ‘customer_type_description’存储在一个通用的代码表中，有一代码表的维护程序来建立此表中的数据，此时客户代码就成了一个OOI.

“读后修改”的案例-1

Measures

- 修改一个员工的记录，用户知道员工的姓名，但是不知道其ID，则：
  - 第一个功能处理是根据名字列出员工（列出员工后，用户可能终止操作）
    - E 请求列出员工
    - R 读员工
    - X 显示员工
    - X 出错信息
    - 规模为 4CFP
  - 第二个功能处理为选择员工，并显示需要修改的信息（列出详细后，用户可能终止操作，识别为独立的触发事件）
    - E 选择员工ID
    - R 读员工的信息
    - X 输出员工的所有信息
    - X 出错信息
    - 规模为 4CFP

“读后修改”的案例-2

Measures

- -第三个功能处理为修改(假定忽略需要验证数据的正确性)
  - E 修改员工数据
  - W 存储员工数据
  - X 确认信息或者出错信息
  - 规模为 3CFP
- -总的规模为：4CFP+ 4CFP+ 3CFP= 11CFP

- 用户的每次选择决策都意味着一个新的触发事件，意味着一个新的功能处理。

简单查询的案例

Measures

- 用户输入开始日期与结束日期,系统输出在此周期内的每个客户的订单个数,每个订单上只有一个产品,输出的数据属性有:客户ID,订单个数.
- 该FUR的OOI有3个:
  - 时间区间(开始日期, 结束日期).
  - Order (Order ID, Client ID, Product ID, order date, ...)
  - 查询结果(客户ID,订单个数)
- 该FUR的功能点有3个(忽略了出错信息):
  - E 开始日期与结束日期
  - R 订单
  - X 客户ID,订单个数

稍微复杂的查询案例

Measures

- 输出时要求输出客户的姓名,地址和ID
- 该FUR的OOI有4个:
  - -时间区间(开始日期, 结束日期).
  - -Order (Order ID, Client ID, Product ID, order date, ...)
  - -Client (Client ID, Client name, address, ...).
  - -查询结果(客户ID, 订单个数)

稍微复杂的查询案例答案

Measures

- 该FUR的功能点有5个：
  - -E 开始日期与结束日期
  - -R 订单
  - -R 客户信息
  - -X 客户ID, 客户姓名, 客户地址
  - -X 客户ID, 订单个数

复杂查询的案例

Measures

- 用户输入3个时间段:开始日期与结束日期,系统输出在此3个时间段内的每个客户的订单个数,每个订单上只有一个产品,输出的数据格式为:
  - 客户ID,客户名称,客户地址
    - 开始日期1,结束日期1,订单个数
    - 开始日期2,结束日期2,订单个数
    - 开始日期3,结束日期3,订单个数
- 该FUR的OOI有4个:
  - 时间区间(开始日期1, 结束日期1,开始日期2,结束日期2,开始日期3,结束日期3).
  - Order (Order ID, Client ID, Product ID, order date, ...)
  - Client (Client ID, Client name, address, ...).
  - 查询结果(客户ID, 订单个数)

多级汇总报告的案例

Measures

- FUR:系统中存储了公司的所有的员工信息,在员工信息中记录了每个人属于哪一个部门,系统中还记录了公司的组织结构,每个部门属于哪个大区.要求输出一个报告,按大区,部门分类,列出每个人的姓名,并汇总出每个部门,每个大区以及整个公司的人数.
- 功能点计算;
  - -E 选择报告
  - -R 员工
  - -R 部门
  - -X 员工姓名
  - -X 部门合计
  - -X 大区合计
  - -X 公司合计

Drop-Down list的度量

Measures

- 情形1:列出合法的值,而没有从其他OOI处获取这些信息,可以忽略.
- 情形2”列出合法的值,从另外一个OOI处读取这些信息,则识别为读与输出各一次
- 情形3:系统提供2种方式供用户选择:
  - 客户知道要输入的值,直接输入,而系统检查是否合法,则识别为一个读.
  - 客户不知道要输入的值,先由系统列出合法的值,客户再选择,此时识别为1个单独的功能处理,该功能处理识别为1个输入,1个读,1个输出.

功能复用的度量

Measures

- 如果A调用了C, B也调用了C, A和B是2个功能处理， A, B都在度量范围内, C不是一个单独的功能处理，则C应被度量了2次
- 如果A调用了B, A, B都是功能处理, 都在度量范围内, 则B应被计算1次
- FFP度量的是功能需求的规模, 而不是物理实现的规模.

温度控制的案例

Measures

按一定时间间隔控制温度的功能处理

事件:实时时钟的节拍

1 x Entry (时钟滴答)

1 x Entry (当前的温度)

1 x Read (目标温度)

1 x Exit (加热器开/关)

功能处理的规模是: 4 x CFP

Measures

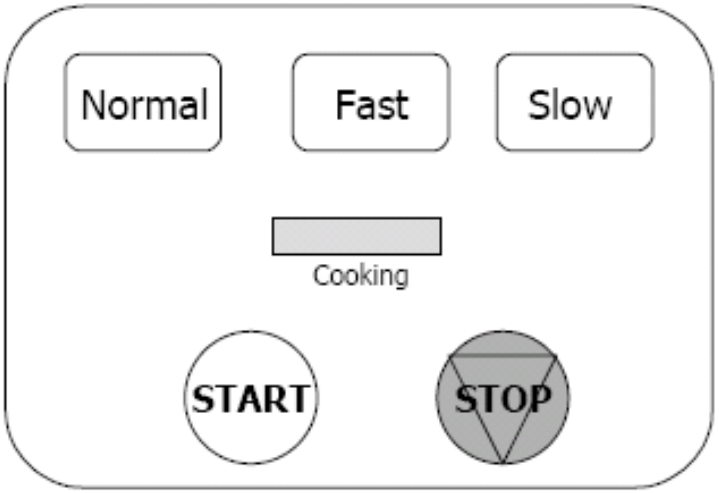
电饭煲的完整案例

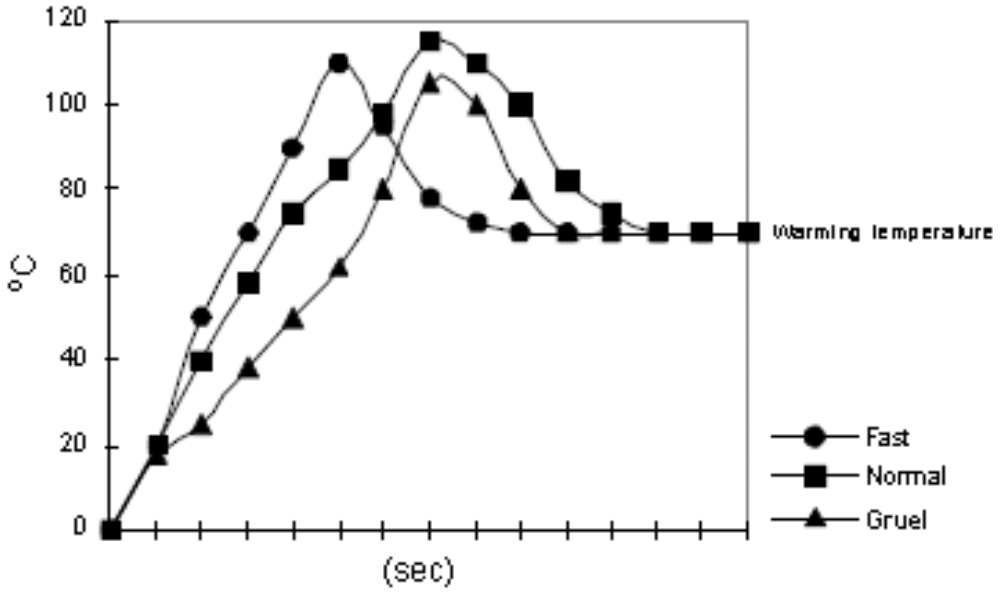
电饭煲的规格说明

Measures

1. 电饭煲可以采用3种模式煮饭：快，正常，慢
2. 当按下START按钮后，开始煮饭.在正常情况下已经选择了煮饭模式.
3. 如果按下了START按钮，而没有选择煮饭模式,电饭煲自动选择正常模式.
4. 当电饭煲完成煮饭后，无论是哪种煮饭模式，电饭煲自动进入保温状态
5. 指示灯会指示出当前是煮饭状态还是保温状态
6. 加热器必须是受控的。软件必须能够按照一定的曲线,为给定的煮饭模式在给定的时间内决定目标温度.
7. 它能提供3种类型信号：流逝的时间，5秒和30秒的循环时间信号。
8. 每隔30秒要重新设定目标温度
9. 每5秒钟，通过用目标温度减去实际温度的值以切换加热器的ON或OFF。
10. 根据流逝的时间和选定的煮饭模式，每30秒修改煮饭和保温指示灯的状态
11. 如果按下了STOP按钮则切断电源

电饭煲的规格说明


Normal
Fast
Slow
Cooking
START
STOP


120
100
80
60
40
20
0
°C
(sec)
Warming temperature
Fast
Normal
Gruel

度量步骤

Measures

- 1 识别度量目的
  - 确定该设备中为上述需求规格所定义的软件系统的规模
- 2 识别范围
  - 上述需求规格所定义的软件功能
- 3 识别层
  - 仅一层
- 4 识别功能用户
  - 发送数据给软件的功能用户
    - 定时器
    - 传感器
  - 接收数据的功能用户
    - 指示灯
    - 加热器
  - 还有其他人吗？
    - 没有了。人是通过开始按钮间接操作电饭煲
  - 还有其他设备吗？
    - 没有了

度量步骤

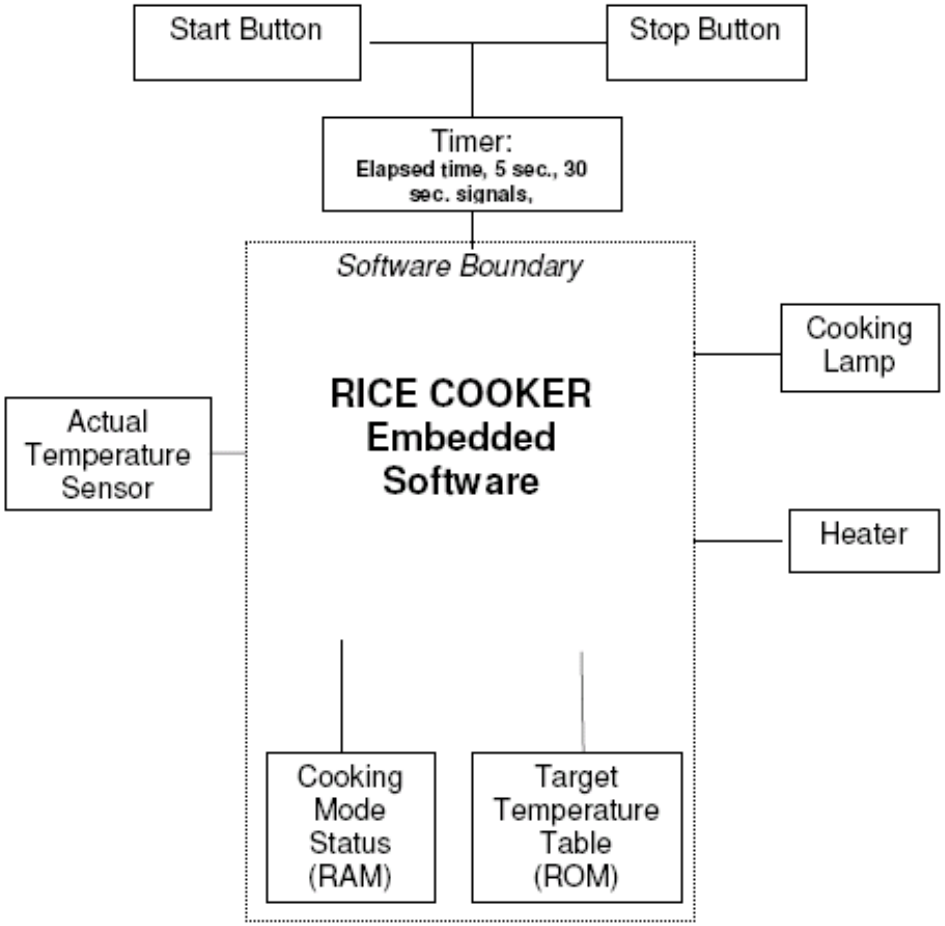
Measures

- 5 识别边界
- 6 识别触发事件

  5秒时钟信号事件.

  30秒时钟信号事件.

  消逝时间

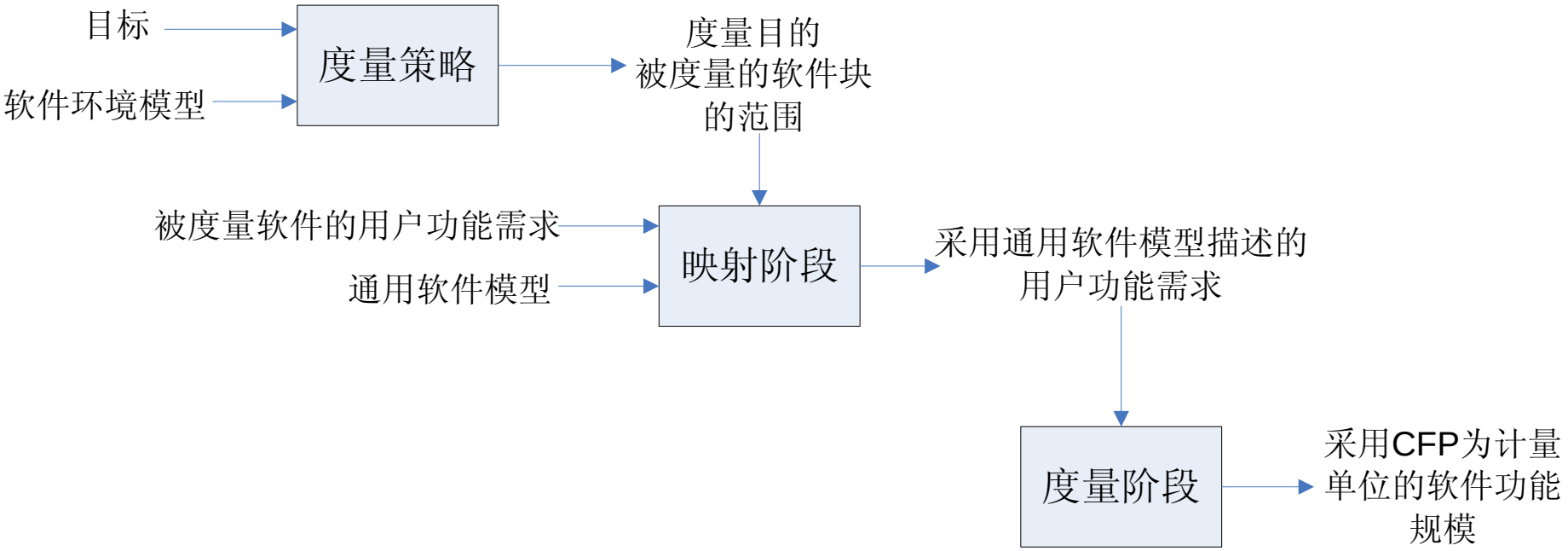


度量步骤

Measures

- 7 识别数据组
  - 煮饭模式
  - 流逝时间
  - 5秒时钟信号
  - 30秒时钟信号
  - 实际温度
  - 指示灯状态
  - 目标温度
  - 加热器状态
- 8 识别功能处理
  1. 控制加热器.
  2. 设定目标温度.
  3. 控制指示灯
- 9 识别子处理
- 10 计算度量结果

小结：FFP的过程模型


目标
软件环境模型
度量策略
度量目的
被度量的软件块
的范围
被度量软件的用户功能需求
通用软件模型
映射阶段
采用通用软件模型描述的
用户功能需求
度量阶段
采用CFP为计量
单位的软件功能
规模

参考资料

- The COSMIC Functional Size Measurement Method Version 3.0(The COSMIC Implementation Guide for ISO/IEC 19761: 2003) September 2007
- Guideline for sizing business application software using Cosmic-FFP ,version 1.1 , May 2008

Measures

谢　谢！