软件 规范

计算机科学1993Vol.20№.4

69-72

软件规范

TP31

黄林鹏 孙永强 倪德明 童维勤
（上海交通大学计算机系 上海200030）

摘 要

规范（specification）的目的是提供一个标准，用以引导和评价软件系统的设计、实现和维护。本文从术语"规范"的定义出发，讨论一个良好的软件规范必须具备的性质，分析了形式规范、半形式规范和非形式规范的优缺点，给出了一些典型的规范系统的分类及相互间的关系。

一、引言

Boehm认为，一个软件的开销大概有二分之一甚至三分之二用于维护，因此减少软件开发费用的任何尝试都应注重软件维护的开销。一般降低软件维护需要的主要途径有：a. 降低矫正的需要；b. 降低修改的工作量；c. （通过使用标准构件）减少总的开发规模。一个良好的规范对所有这些途径都有着重要的影响。因此为了改善软件的质量和提高总的生产率，我们不应减少规范工作，而应增加投入，由此在维护上（同时也在设计、实现及集成上）节省更多的开销。

在软件工程领域也许没有其它任何一个词比"规范"有着更多的定义。当"规范"用作一个名词时，它和词"文档"有着相似的含义，如一个需求规范即是一个记录需求的文档；当"规范"用作形容词时，它可修饰软件编码之前的任何工作，如"规范阶段"广泛用于表述关于系统所有模块的外部可视行为的定义和文档活动（产生了一般所称的"模块规范"、"低层设计"或"详细设计"），或者用于表述整个系统的外部可视行为的定义（产生了一般所指的"软件需求规范"）；当然也有人使用"规范"这个词来表示软件的任何（不考虑抽象层次的）形式描述。

尽管"规范"的定义有多种，但这里我们

这使得分析模型得到增强。

（3）嵌入对象的概念使得对象在与其相连的父子关系中得到标识。嵌入对象的概念建议了一个重要的观点：通过构造对象行为的有序集，来使得标识隐含在其它对象结构中的对象得到标识。

（4）通过考虑对象之间的关系，可标识一对象与其它对象之间的组合关系。

（5）在f(JSD)中引入分类的概念以增强其模拟现实领域对象的能力。

（6）不属于任何模型对象定义的功能需求应与对象模型的定义分开，且作为一个服务独立地指定。在高层抽象中，它们可看作是属于伪对象方法的一个伪对象。

（7）最终的系统规范是由模型对象和伪对象的集合组成。

综上所述，f(JSD)加上分类的概念及其继承特性构成对象模型和功能需求模型，两者一起便可作为OOM的第一个步骤——OOA应用于信息系统开发中。

主要参考文献

[1]诸葛海 等著《信息系统与类比方法论》，浙江大学出版社，1992。

[2]诸葛海 编著《信息系统开发方法与环境》，浙江大学出版社，1993。

给出的是IEEE关于"规范"的定义：

一个规范是(1) 一个以完备、精确、可验证方式描述的一个系统或其组成成份的需求、设计、行为或其它特征的文档。参见设计规范、形式规范、功能规范、界面规范、性能规范、需求规范；(2) 开发一个规范的过程；(3) 一个关于一个产品、一个材料或规程表示所满足的需求集合的简明陈述。一旦合适，也包含一个由其可判定的给定需求是否被满足的过程。(ANSI N45.2.10)

二、规范的组成、目的和性质

1. 规范系统的组成和一般结构

当谈及一个规范系统时，我们一般区分其三个组成成份：方法、语言和工具。方法表示规范是如何进行的；语言通过语法表示限制了可用的语句集合；而工具提供了检验、存储和传送这类语句的手段。方法、语言和工具通过规范系统所涉及的抽象概念紧密相关。

下面我们列出这些成份所期望具备的性质：

a.抽象概念

- 逐步完备。
- 永久有效。

b.由系统支持的方法

- 快速加入每一信息。
- 允许非形式文本。
- 尽早检查正确性、完备性、一致性、无二义性。
- 集中于规范所必需的信息。

c.语言

- 具备一个规范的数个语法表示（如图、表等）。由于规范由多人，包括系统分析员、用户、管理人员和程序设计者等书写和阅读，其背景和兴趣各不相同，所以他们对规范的内容和风格存在不同的要求，因此有时需要一种工具来自动转换不同形式的规范。

d.工具

- 提供版本及其变种管理的多用户数据库系统。软件系统可能由多人参与开发，因此同时会存在多个软件版本或变种，上述工具的需要是显然的。
- 检验、恢复和选择的工具。

然而，在现实中大多数规范系统不是完备的，通常只具备上述成份或性质的一部份。

2.规范的目的

规范的目的是提供一个标准，用以引导和评价软件系统的设计、实现和维护。该标准除了包括系统必须能做什么的清晰定义（需求规范）外，还包括下述约束：

技术上：为了创建和操作，系统必须使用何种技术；

操作上：何地且如何使用系统；

生产上：开发系统使用何种方法；

组织上：系统如何适合目标组织；

社会上：系统如何影响它的操作环境。

规范过程是为了在系统生存期过程中使用、说明和记录这些信息。换言之，规范过程必须既能"询问正确的问题"，也能"记录正确的信息"。

3.良好规范的性质

一般我们要求一个良好的规范应是：

(a) 正确的，应反映实际的需求；

(b) 完备的，应包括所有相关的需求；

(c) 一致的；

(d) 无二义的；

(e) 防止信息丢失和意外的改变；

(f) 易写、易修改；

(g) 易读且简明，便于用户和分析者间的通讯；

(h) 可实现，应易于设计和实现；

(i) 可验证，应存在一个过程来检验产品是否满足它的规范；

(j) 有效的，应有一个机制来保证该规范确实反映了用户的规范；

(k) 可跟踪，即当规范改变时，应易于在其它文档中确认受该改变影响的所有语句。

要注意的是，上述目标可能相互矛盾，

如一个形式规范（代数规范）是可验证的，但对大多数人来说不易阅读，可能不是有效的，难以确认规范确实反映了用户的目标。进一步，上述的前四个性质不是对所有人都有相同的含义，例如，规范工具的提供者一般都宣称其系统能保证正确性，然而这并不意味规范的内容对顾客的意图是正确的，事实上，指某种形式规范被满足。其原因在于除了用户的大脑外没有任何参照能证明规范的正确或完备。相反，程序关于其规范的正确是可证明的。

三、规范的形式化层次

在IEEE中形式规范的定义为：(1) 一个依照已建立的标准书写和证明的规范；(2) 在正确性证明中，一个系统或其成份的外部可视行为的形式语言描述。

形式规范语言为了实用，一般需要某种验证能力，但软件开发者所特别关心的是形式规范语言适用于哪些应用领域及其易读易用性如何。

与形式规范语言相比，非形式规范语言在多数情况下易读易用，但也产生了多义性问题，使规范的内容是否符合规范的意图变得更加复杂。

为了更详细区分规范的形式化层次，J. Ludewig引入了另两种形式化层次：格式和半形式规范，而把"形式"规范限于形式推理中可使用的语言，把"非形式"规范看成是一般不受任何限制的自然语言。这四种层次的语言可用表1加以区分。

表1　不同形式化层次规范语言的比较

风　格	语　法	语　义	例
非形式	非精确定义	非精确定义	自然语言
格式化	限　制	非精确定义	表格
半形式	定　义	部份定义	伪码
形　式	定　义	定　义	顺序语言

一般，我们使用一个形式语言编制程序，使用非形式语言记录文档，有时也附加表格加以说明。表格限制了自然语言使用的方式，它要求用户回答所有的相关问题。半形式规范语言的一个典型例子是伪码，是把控制字嵌入自然语言所得到的一种程序的设计语言。

计算机理论工作者所研究的形式规范技术，如代数规范，出于某些原因未能得到充分的实际应用。但基于半形式语言的半形式规范和形式规范相比却具有下述优点：

(a) 易于没受过形式方法全面训练的人学习和理解；

(b) 它的文档类似于自然语言所书写的文档，便于不同背景的人使用；

(c) 非完备和模糊信息在这样的系统中能得到良好的表示。

另一方面，半形式规范和非形式规范相比，具有如下优点：

(a) 使普通正文中潜在的许多缺陷变得

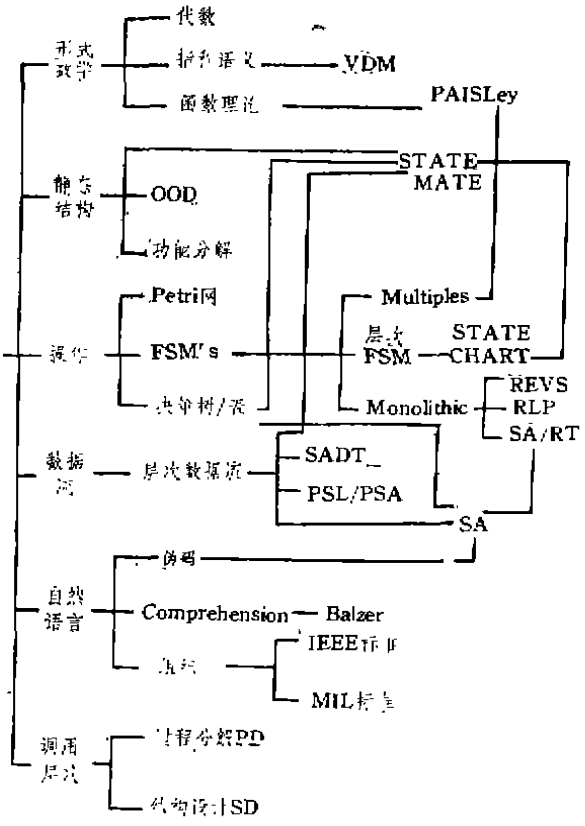

图1　规范谱系树

可见；

(b) 它可在一个数据库中进行存储和更

正；

(c) 可使用自动工具来检验和改变记法。

因此，半形式规范已得到越来越多的关注和使用。

四、规范系统分类及相互间的关系

一般规范系统可分为六个主要类别（图1）：形式数学规范、静态结构规范、操作规范、数据流规范、自然语言规范和调用层次规范。下面作一简单讨论。

形式数学技术可分为三个范畴：代数、指称语义和函数理论。代数方法使用形式逻辑和集合论理论，定义在合适条件或输入状态下系统或其成份的输出状态；指称语义一般用于定义语言翻译并允许我们形式规定所有语言结构的操作语义；函数理论方法把软件的行为规定为从输入到输出的数学函数。

静态结构方法使得我们能把系统的一个部份规定为系统的另一个部份。它一般（但并不总是）以自顶向下的方式使用，而且是许多高级设计方法的主要基础。

操作方法以一个可执行的方式描述软件的外部行为。该方法一般先假定一个模型，如有限状态机(FSM)、Petri网、决策树/表，然后把对软件外部行为的规定看成这些模型之一的一个设计。在有限状态机方面至少已有三个不同的发展：单片FSM，即整个系统作为单一FSM描述；多合作FSM，即FSM间通过消息传递进行通讯，层次FSM，即FSM可被分解成相互合作的FSM的集合。

数据流方式使我们能规定信息通过一个相继数据传送网进行流动和处理。

自然语言方法使得用户能使用自然语言说明软件，一般可使用文档标准来帮助把文本组织成有意义的段。目前已有一些技术可用来把非形式书写的规范转换成更形式化的说明。伪码(Pseudocode)规范使得我们可把一些控制字如IF, DO WHILE, CASE等嵌入到自由形式的语言中，因此有些人把它称为半形式规范。

调用层次方法是一个用得最早的方法，它的特点是定义主程序，然后再定义由其调用的子例程。

五、结束语

经过多年的努力，现在规范系统的开发者已逐渐了解到，规范系统的理想观点是不存在的。许多复杂系统要求其规范者从不同的侧面来考察问题和求解。因此，目前的一些规范系统都提供了多种方法。如在SA/RT中，既可把系统规定为层次数据流图和控制流图，也可把系统规定为FSM或决策表。在STATEMATE中，既可把一个系统规定为层次数据流图，也可规定为层次FSM或层次静态构件。我们认为，一个成功的规范系统一定是一个能提供从各个侧面观察和描述问题并以多个表示形式加以规定说明的系统。当然一个把所有侧面和多种表示自动集成起来以形成一个单一相关的系统模型的工具也是必不可少的。

参考文献

[1] Ludewig J., Languages, Methods, and Tools for Software Specification, in Zalewski J. and Ehrenberger W(Eds), Hardware and Software for Real Time Process Control North-Holland, 1989

[2] Carl K.C., Specification Languages-Assessment and Trends, in Inter. Conf. on Computer Languages, 1988, 160

[3] Paul C. G., A Perspective on Specification Languages (as above), 164

[4] Berg H. K., et al., Formal Methods of Program Verification and Specification, PRENTICE-HALL, INC, 1982

[3] Li Wei, Requirement Techniques, DeTfoRS 1st Workshop, 1992