计算机科学1990№.5

JSP & JSD

蔡建明　林钧海

（南京航空学院）

摘要：本文全面介绍了Jackson 方法的主要思想及其形成过程，并对该方法作了简要评述。同时，针对国内研究中的有关提法进行商榷。最后，谈到了支持Jackson方法的软件开发环境。

一、概　述

Jackson 方法是以数据结构为基础开发软件的代表性方法，它由英国的M．A．Jackson 提出，经过若干人十几年的努力，目前已成为一种有影响的软件开发方法。它的发展分为二个阶段，前期（70年代）主要研究结构化程序设计，称为JSP （Jackson Structured Programming），后期（80年代）集中研究软件系统的开发，称为 JSD （Jackson System Development)[3]。国内不少软件工程专著称JSP为Jackson方法的提法值得商榷，时至今日，Jackson 提出的方法已不仅仅用来设计程序，而且是开发软件系统的完整方法。

与传统的功能分割，结构化分析和设计的观点不同，Jackson强调对问题解的组合而不是分解。Jackson坚持认为[2]，自顶向下逐步求精的方法只适合描述问题的解，并

83a]，以取代通常独立于语言的机制。其思想就是程序可分解成任意的片段(模块)，一个片段可以是由文法的非终结符产生的符号和文法的终结符所组成的字符串，通过定义片段间的偏序，可以进行独立的上下文有关（静态语义）分析。这种方法可将一个模块分成接口部分和实现部分，从而提供对数据机制的保护。最后，这种机制可作为构造处理不同程序版本/变体的系统的基础。

• 对一大类系统例如操作系统和数据库系统来说，在系统的寿命（执行）期内交换部件的能力是系统固有的和自然的特性。为了做到这一点，必须有一个包括程序执行和完成该程序执行的处理器的机器模型。目前，BETA正在研究这种模型，机器将被看作是按照某种语言描述的剧本（程序）实现（程序执行）的一个整体。

• 实现一个给定应用领域中的概念和现象的一组属性（模式和引用）可看作为描述属于该应用领域的系统的语言。更为常见的要求是使用一种语法而不是BETA来描述这样的系统，因此，BETA程序设计系统应该包括一个工具，给定一组属性及其语法描述，该工具将产生一个新的BETA 编译器的前端。

尽管上面引出了一些BETA 有待改进之处，但我们认为BETA 已达到可用来编制计算机系统程序的水平。BETA 的一个实验版本已经在各种机器（包括Macintosh，Sun和Syswar SUS）上实现。目前，有关设计和实现一个BETA程序设计环境〔Mjølner〕，〔Scala〕的工作正在继续。

参考文献

1. B．B．Kristensen et al,The BETA Programming Language, Research Directions on Object-oriented Programming, The MIT Press, 1987．pp．8-48．

不适合开发未知的解。Jackson方法采取了与自顶向下截然不同的开发路径。在JSP中，先分别画出数据结构，然后将其组成一个程序轮廓。在JSD中，先产生规格说明，在实现时仅仅作变换而不是求精。因此，Jackson方法不是一种自顶向下的软件开发方法。

二、JSP

JSP是一种设计程序方法。JSP的开发从输入/输出数据出发说明一个不完全，不精确的问题，结果是一个结构化程序。JSP用数据结构图构成要解决问题的模型。Jackson提出的数据结构乃至程序结构都仅限于三种基本逻辑以及它们的复合。实际上这也是结构程序设计原理中的过程性结构。JSP采用Jackson定义的结构图以及等价的图式逻辑(Schematic Logic，也称为结构文本)。JSP并不限定某种程序设计语言编码及通常列出的可执行操作即编码所用程序设计语言的可执行语句。JSP还提供了许多有用技术以便使之适应各种复杂的程序环境，其中包括：解决结构冲突，出错处理及回溯技术以及程序转换等。许多专著及[5]已有详细介绍，这里不再赘述。

三、JSD

随着问题规模的增大，输入输出数据流之间结构冲突的可能性亦随之增大。尽管利用中间文件解决了冲突，但它明显降低了功效，而且从概念上说中间文件也是不必要的。由此我们可以看出，JSP并不适用解决大型软件的开发问题。

针对JSP遇到的困难，Jackson在70年代末提出了JSD的思想，而JSD的发展与完善则是Jackson和J. R. Cameron共同完成的。1983年Jackson和Cameron分别介绍了JSD[2,3]。目前，尤其在欧洲，一些商业组织已使用JSD成功地开发了一些应用软件系统。

JSD为被开发的软件系统产生规格说明并且实现它。下面列出JSD的开发步骤以及每一步所做工作。

(1) 实体动作步(entity action step)
列出系统涉及的实体和它们的动作，定义与之有关的客观世界范围。

(2)实体结构步(entity Structure Step)
用Jackson的结构图描写每个实体所执行或承接的动作，其先后次序在结构图中表明。

(3)初始模型步(initial model step)
将前二步形成的系统描述中每一客观实体及其动作序列模型化为顺序进程。通常称这些进程为模型进程，由它们构成模型。同时，通过模型与客观世界之间建立的数据通讯达到模型与客观世界行为上的同步。

(4)功能步(function step)
用JSP技术开发功能进程。根据开发的需要增进模型进程的内容，以使系统产生用户所需的输出数据。

(5)系统性能步(System timing Step)
非形式化的定义系统输入输出的各种性能要求，在下一步实现系统时必须满足这些要求。

(6)实现步(implementation Step)
使先前产生的规格说明适应目标计算机软/硬件环境，使它足够有效地运行，并满足系统的各种性能要求。实现的结果是最终软件系统的程序代码，JCL(Job Control Language)或者操作指令。

上面六个开发步骤中，(1)至(5)产生完整的规格说明，仅用第(6)步实现规格说明。

JSD的规格说明由一些顺序进程构成。定义单个顺序进程的符号即JSP中的结构图和结构文本。JSD中用SSD(System Specification Diagram)表示各类进程间的通讯联系。JSD在建立规格说明时采用了计算机仿真的方法，先研究系统有关的实体、属性和它们的活动，再建立系统模型。现实世界中各个独立的活动实体以进程的形式出现在模型中，一个进程可以有局部变量，它描写实体的属性。JSD的规格说明中每个进程被认为是在一个特定的处理器上执行。因此，没

有进程的调度，也没有处理器的共享问题。每个进程属于一个进程类，一个进程类有一个程序文本。

传统的目标机其硬件环境只有单个处理器。因此，给定一个足够速度和存储能力的处理器后，使规格说明在该硬件环境下运行是实现阶段的基本任务。实现的本质是提供一个通用的操作系统，利用单个处理器调度各个进程的执行以及管理数据流队列[4]。

把规格说明转变为单处理器环境下运行的软件系统主要采用三种技术：变换，调度(Scheduling) 以及状态向量的存取。最重要的变换是程序转换 (Inversion) 和状态向量分离 (State Vector Separation)，变换的结果用SID (System Implementation Diagram) 表示。进程调度是操作系统十分重要的工作，一个通用的调度程序可能是低效的，但十分专用的调度程序又可能不易改变。可以有多种方法实现调度决策。第一种方法是利用Ada的任务 (task) 功能，将调度决策保留到运行时实现；第二种方法是通过转换及调度程序在JSD的实现阶段确立调度决策；第三种方法可以利用并行程序在实现阶段确定调度决策。JSD 采用第二种方法。不同的调度规则得到不同的软件系统（实时响应，批处理等）。同类进程中的各个进程通过相同的程序文本及各个进程各自的状态向量实现运行，这就是状态向量分离的基本思想。通常，状态向量放入文件系统或数据库，实现阶段必须确定文件或数据库的设计，以便各个进程能够以可接受的效率存取状态向量。

限于篇幅，有关JSD的实例参阅[2,3]。

四、对Jackson方法的认识

JSP适合小型程序的开发，是一种结构程序设计方法。随着问题规模增大，JSP中用输入/输出数据流描写问题的方法越来越不适用。同时，结构冲突的可能性及解决冲突的算法复杂性亦随之增大。JSD 正是为解决这些问题发展起来的。不要认为JSP与JSD有明确的界面，或者认为JSD是JSP的前期工作，事实上JSP的技术已经渗透到了JSD中。

JSD并不明确区分分析与编程的界线，其中只有规格说明和实现之分。JSD将系统功能的考虑推迟到开发的后期。其基本原则是开发者必须为现实(reality)构造模型，这个模型代表了客观世界中和系统有关的现实，系统自身可看作对这些现实的模拟。通常，模型比功能更加稳定。JSD中模型先于功能的观点对系统的开发与维护都是非常有利的[6]。同时，对开发者与用户来说，基于用户的客观世界通讯比基于开发者的技术领域通讯更为有效。

软件开发的现实目标是将开发任务分解成许多经过良好定义的步骤，使开发者确信每一个开发步骤都朝着满意的解迈进。开发的每一步都要对被开发的系统作出许多决策。如何组织开发过程的各个决策可用来刻划一个方法的特征。下面几点说明 JSD是一个好的开发方法。

(1) 作出容易的决策应该先于困难的决策。JSD中第一个决策是关于客观世界的，它提供了系统的主题 (subject) 内容。系统预期的用户心里早已有一个客观世界的模型，并且能够对开发者在前二个步骤形成的抽象描述给出权威性的评判。

(2) 应当尽可能推迟最易于导致错误的决策。有关要开发的系统的决策是对一个尚未存在的东西的决策，因而最易于导致错误。JSD使这些决策推迟作出，使有关系统的规格说明推迟到功能步作出，使实现的决策推迟到开发的最后作出。

(3) 含蓄的决策应当避免；开发者应该清楚每一步要决定的内容，不应有隐含的约定存在，否则到开发的后期会招致困难，并且可能无法克服。传统的开发方法在功能规格说明中隐含着现实的模型，在设计实现中隐含了强制的功能规格说明。JSD 在功能规格说明之前对现实模型作出了明确的决策，先于设计实现之前作出了功能规格说明的决

策，从而避免了这些隐含约定。

（4）如果一个决策易于出错，应当尽早证实它的正确性。JSD中，实体结构步能立即检测实体和动作列表的正确性。这些结果在功能步又得到检测。关于功能进程的规格说明可以检测通过某种方式将此进程连接到SSD中决策的正确性。

（5）应当尽可能使决策之间没有依赖关系，使它们互不相关。JSD中描述实体和它们行为的现实是非常松散地联系起来的，通常可以十分容易地改变其中一部份描述而不影响其它描述。在大部份情形下，功能仅仅与模型连接，而不是互连。因此，它们可以被独立地说明。

采用自顶向下逐步求精（TDSR)的思想构成了软件开发方法的一个主要流派。有不少阐述软件开发方法学的专著实际上都是介绍TDSR的，从中使我们领略到TDSR的影响之大。但是，随着软件开发学科的发展，人们越来越多地发现TDSR的缺点与缺陷，而Jackson方法却在TDSR的许多不足之处显示出它的优越性[7]。

Jackson方法存在以下三个方面的问题：

（1）Jackson方法中并不包含诸如选题，计划管理，代价/收益分析等内容，也不包含实施环境问题的研究内容，这在一定程度上影响了它的推广应用。

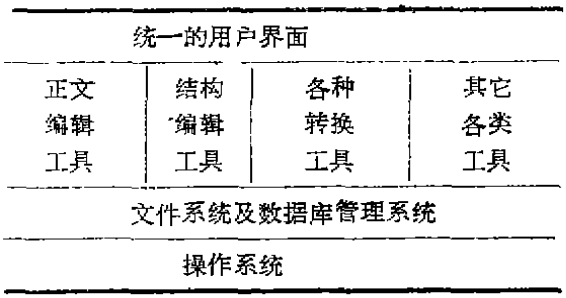
（2）JSD中并未揭示出算法和系统输出流与客观世界的内在联系，这使功能进程的开发仍基于JSP直接进行，并使得软件系统中功能进程的组织由开发者随意安排，这样导致了模型进程和功能进程概念与结构上的不一致，给开发工作带来一定的困难。

（3）JSD在实现阶段所作的变换十分费劲，以致利用人的智力进行变换难于保证其有效性和可靠性。而且，变换的选择又过于复杂，目前仍无法用计算机自动实现。

五、关于开发环境的考虑

目前，支持TDSR方法学的软件开发环境已较成熟，但支持Jackson方法的软件开发环境尚未见报道。我们认为，Jackson方法是一种实用的软件开发方法，如果没有集成化的开发环境支持，再好的方法都只是空中楼阁。

借鉴现有成熟的开发环境构造技术[10]，我们设计Jackson方法的开发环境结构如下所示：



开发环境应满足以下七个方面的要求：

（1）JSD比JSP更适合于开发大中型的软件系统，开发环境应重点支持JSD并兼顾JSP。

（2）开发环境应是开放式渐增的系统，以便将各种工具（包括目前实现阶段还无法自动化但今后可能产生出的变换工具）有机地结合进开发环境。

（3）结构图和结构文本是一一对应的，必须有专门工具进行相互转换。

（4）应有支持文字和结构编辑的工具，人机交互地产生用户的结构文本及非形式化的描述。

（5）应提供方便的结构图形编辑工具，帮助用户产生SD、SSD及SID。

（6）作为开发环境，各个工具应是密切相关的集成化整体。各个工具产生的信息应通过文件系统及数据库统一管理，使各个阶段的工作相互衔接。例如，在实现步能够方便地查询和显示功能步得到的SSD及系统性能步产生的对实现的非形式化约束。

（7）应尽可能提高系统的适用性，如增加方便初学者使用的help功能，命令挽回功能等。

作为先期的验证工作，我们已在IBM

计算机科学1990 №.5

关于编译程序测试的若干问题

李友仁 （西安交通大学）

摘 要

In this paper we have analysed and researched Objective, principles, design and implementation method for compiler testing. This will implement and improve the theory and implementation methods for compiler testing.

一、编译程序测试的目标和特点

对软件系统进行测试的主要方法是通过设计测试实例，执行程序和分析运行结果来发现软件系统的错误。编译程序测试的方法与上述方法相同，但编译程序测试的依据是语言标准文本，而不是通常的软件规格说明；测试实例是由一组源程序组成，而不是一组数据；每完成一个测试都需要执行编译-装入-执行(compile-load-go execute)过程。因此与一般软件系统测试不同，编译程序测试有自已特殊的要求和方法。

编译程序测试的目标是要验证编译程序是否遵守语言标准文本的规定。换句话说，它是要检查编译程序与语言标准文本规定的语法、语义和语用的符合程度。通常，编译程序的测试是为了确认编译程序的正确性，

PC/AT上实现了其中的结构编辑功能[9]。

六、结束语

Jackson方法中的JSD方法是一种很有潜力的软件开发方法，它在规格说明阶段并不关心硬件环境，只在最后实现步才确定目标机的环境，并施于各种变换以适应系统环境。随着硬件环境的发展，在并行多处理器系统环境下开发大中型的应用软件系统，JSD有它的优越性。对进程的调度规则作深入的研究，JSD的规格说明可在任意个数的处理器支持下运行。

主要参考文献

〔1〕M. A. Jackson, Principles of program design, Academic Press, 1975

〔2〕M. A. Jackson, System development, Prentice—Hall, 1983

〔3〕J. R. Cameron, JSP & JSD, The Jackson approach to software development, IEEE Computer Society, 1983

〔4〕M. A. Jackson, Ioformation Systems: Modelling, Sequencing and Transformations, Proceedings of the Third International Conference on Software Engineering, 1978

〔5〕苏建文、潘锦平，Jackson 设计方法介绍，计算机科学，1984.1

〔6〕胡绮峰，吴湛兵，陈友君，JSD——一种系统开发方法，计算机科学，1986.1

〔7〕仲萃豪、叶农，JSD方法对软件工程研究的启示，计算机科学，1988.6

〔8〕蔡建明，一种面向数据结构的软件设计技术，南京航空学院科技报告，1988.11

〔9〕蔡建明，Jackson方法及其开发环境的研究，南京航空学院研究生硕士学位论文，1989.4

〔10〕董士海，计算机软件工程环境和软件工具，科学出版社，1988