计算机科学2000Vol.27№.1

脚本语言发展研究

Research of Scripting Languages Developing

邵 莹 刘宗田

（合肥工业大学计算机与信息学院微机应用研究所 合肥230009）

Abstract With rapid increasing in computer speed and changing in mixing applications, scripting languages have been used more and more widely. This report compared scripting languages with well-known high-level programming languages, and specified their features and suiting areas in full. Scripting languages are designed to glue applications together, achieving higher-level applications. It is more convenient programming languages than high-level languages.

Keywords Scripting, Gluing, Typeless, GUIs, Internet, Interpreter

一、引言

在过去30年里，编程人员在编写程序的过程中更多注重的是如何使用高级程序设计语言编写出具有个性的应用程序，但这一观念目前正逐步发生了根本性的变化。这种变化表现为由高级程序设计语言（例如C或C++）向脚本语言（例如Perl[1]或TCL）的过渡。因此，比较脚本语言和高级程序设计语言的各自特点有利于我们了解这种变化的必然性。

脚本语言与高级程序设计语言不同，各自适用于不同的任务。高级程序设计语言是从最简单的计算机基本元素例如内存字开始构造数据结构和算法，而脚本语言是以构件"粘贴"的方式进行设计，假定已经存在有一组强有力的构件，主要任务是怎样将这些构件连接起来。高级程序设计语言是一种强类型的用于复杂处理的语言。而脚本语言是一种无类型的只需在构件之间简单地建立连接实现快速应用开发的工具语言。

脚本语言和高级程序设计语言的关系并不矛盾。两者具有互补性。60年代以来的大多数主要的计算平台都包括这两种语言。构件框架集成是这种互补性的典型代表。它的一般形式是首先使用高级程序设计语言建立构件，然后再使用脚本语言将构件粘贴在一起。随着近年计算机及其相关技术的发展，例如更快的机器、更好的脚本语言、图形用户接口（GUI）的普遍使用、以及internet的高速发展，大大加速了脚本语言的应用。这种发展趋势将会持续到下个世纪，到那时会有更多更新的完全使用脚本语言编写的应用程序，而高级程序设计语言仍将是构筑构件的主要语言。

二、脚本语言

在运用脚本语言设计应用程序的过程中，一般来说只需了解构件的外部特性，无需知识构件内部复杂的算法和数据结构。因此脚本语言有时又被称为粘贴语言或系统集成语言。

1. 脚本构件粘贴

运用脚本语言设计应用程序的过程本质上是一种粘贴过程，即是将符合粘贴条件的构件拼装在一起形成应用。参与粘贴的脚本构件的基本条件如下：1）软件构件之间自由相关，因而几乎不需要预先假设与它相交互的有哪些别的软件构件，且构件可以被简单地重用。2）编程者以"黑盒"方式使用构件，即只能示例和使用它们，不能对其修改、扩展，也不必了解它们的内部执行过程。为了满足这两个条件，使用语言解释器作为可重用构件的中介。解释器能够根据不同的构件类型完成命令解释、示例构件、和控制交互等操作。

构件之间的交互一般采用两种方法：粘贴代码方法和配置脚本方法。粘贴代码交互方法主要完成构件和构件之间的交互。其特性为：构件之间松散偶合；即构件之间的相互依赖性很小；构件可重用，能在尽可能多样的类型环境中使用它们；构件在运行过程中被动态地组合成应用；构件能够被那些甚至没有基本的编程技能的编程者使用。

使用粘贴代码方法需要较多的内存，且为了完成

邵 莹 硕士生，主要从事基于Web的软件重用技术研究。刘宗田 研究员，博士生导师，主要从事软件工程与环境、人工智能、反编译等方面的研究。

每个构件到构件交互操作会带来额外系统开销，因而执行效率低。对于带有图形用户接口的应用程序，或是那些需要在构件之间传送大量数据的场合不能使用这种方法。

配置脚本直接方法主要用于设计构件和解释器之间的交互。从构件的观点来看，解释器也可以视为一组专门处理构件之间交互参数的构件集合，所以构件和解释器之间的交互，也是一种构件和构件之间的交互。但由于这种交互在相互之间需要大量的数据传送，因而不能采用粘贴代码方法。必须采用效率高且需要内存少的配置脚本方法。具体方法如下：

·为构件定义二进制接口。

·将脚本层次的"handles"直接指向一个二进制构件接口，实现一个构件对另一个构件的解释。

·脚本命令的传送是通过示例脚本且在脚本之间传送接口句柄来实现，构件之间通过相互包含指向对方的二进制接口句柄实现交互。

采用配置脚本方法，构件之间不是松散偶合关系。编程人员必须细致设计构件之间的潜在交互和它们在这些交互中所遵循的规则。如果使用 C++ 设计，这些规则和交互能够使用抽象数据接口来定义。如果在 C 中实现，它们能够使用函数指针和返回值定义。

整个脚本语言体系的实现是这两种方式的结合。一般使用粘贴代码方法进行相互独立构件之间的灵活交互。而构件与解释器之间的交互采用配置脚本方法实现。例如，文[7]Tk 的 Button 工具使用粘贴代码方法去指示当用户按下按钮时发生什么事件和通过一个二进制接口句柄去向它的图形管理者发出请求。

从某种意义上说采用高级程序设计语言的程序设计也是一种构件的粘贴，只不过此时的构件被看作是高级程序设计语言中的对象或函数。此种构件之间的关系紧密，且这种关系是编写程序时预先设定好的，并在程序完成时固定下来，不能动态地改变。设定这种关系需要对整个目标应用及实现工具有着透彻的了解，与脚本构件的粘贴存在着本质上的不同。

总之，一个脚本构件的使用应不依赖于应用中的别的构件。构件和构件之间的交互行为不是固定的。它能由脚本编写者参数化，并在执行期间动态地改变。由于粘贴代码方法必须解释执行，用脚本语言编写出的脚本必定会为这种灵活性付出性能上的代价。

2. 脚本语言的特点

由以上脚本构件的灵活的粘贴特性的要求，脚本语言必须具有如下特点：

(1)脚本语言的无类型性。为了粘贴构件任务的简易性，脚本语言要求是无类型的。例如 Tcl 和 Vb Script 中的一个变量在某一时刻保存有一个字符串而在另一时刻可能保存有一个整数。脚本语言对于构件的形式没有预先的限制，而且所有的构件和值都以统一的形式表示。因此，任何构件或值可以在任何场合使用。为了某一目的设计的构件可以用于设计者没有想到的完全不同的目的。例如，文[2]在 Unix shell 中所有的过滤程序从输入读入一个字节的流然后写一个字节的流到输出，任何两个程序可以通过这种方式连接起来。下面的 shell 命令叠放三个过滤器在一起以计算包含有字"scripting"的行数：

```
select|grep scripting |wc
```

select 程序读取当前显示中被选择的文件并且输出文件到输出端，再由 grep 程序读入并且对于包含有"scripting"字符串的行输出，并由 wc 程序计算输出行的行数。这些程序中的任何一个都可以应用于执行其他不同任务的多种场合。

由此看来，脚本语言的无类型特性似乎允许某些错误存在，但是实际上脚本语言与高级程序设计语言是同样安全的，对于参数错误检查，脚本语言一般是在运行时进行。强类型高级程序设计语言是在编译时进行，因此在运行期间检查所需的时间花费被节省下来。这样以效能为先决条件的结果限制了信息的使用，同时减少了程序的灵活性。

(2)脚本语言的被解释执行。脚本语言和高级程序设计语言的另一点不同是脚本语言通常是被解释执行，而高级程序设计语言是编译执行。解释执行不需要花费编译时间，能够快速地实现应用。由于允许用户在运行期间编制应用程序，解释器还可以更加灵活地粘贴构件。例如，某些集成电路的综合分析工具就集成有 Tcl 解释器。另外解释器还允许通过以动态产生代码的方式使得实现功能的工作更加有效。例如，一个基于 Tcl 的网络浏览器能够通过使用几乎没有规则的表述代替翻译主页的 HTML 成为一个 Tcl 脚本语言，然后执行脚本语言将页显示在屏幕上[3]。

脚本语言比起高级程序设计语言来较为低效，部分原因是由于它们使用解释器代替了编译器，也有部分原因是为了使它们的基础构件更加有效而对其进行了封装，使得其更加易用。例如，脚本语言可以根据情形的不同使用可变长度的串，而高级程序设计语言必须使用一个对应单个机器字的二进制值，并且脚本语言使用 Hash 表，而高级程序设计语言使用索引数组。幸运的是，性能并不总是主要因素。脚本语言的应用程序通常比高级程序设计语言的应用程序要小，并且使用脚本语言设计的应用的性能也受到其构件性能的影响，而构件的实现完全使用高级程序设计语言。

脚本语言比高级程序设计语言处于更高的层次，一条语句比起高级程序设计语言能够做更多的工作。一条典型的脚本语句能够执行几百或几千条机器指令，而相对来说高级程序设计语言的语句大约能够执行五条机器指令。这些差异的产生，部分原因是由于脚本语言使用了解释器，根本原因还是由于脚本语言中

的原语操作具有更强的功能。例如，Perl 对于规则表达式的引用就象对整数地址的引用一样容易[1]。

三、选用语言的策略

脚本语言不能取代高级程序设计语言，反之亦然。它们各自适用于不同任务。有了粘贴和系统集成，使用脚本语言设计应用程序速度可以提高5到10倍。高级程序设计语言需要大量加工和转换代码来连接不同的部分，而这种连接在脚本语言中是直接进行的。对于复杂的算法和数据结构，使用强类型的高级程序设计语言使得程序更加容易管理。同时对于要求执行速度的应用，使用高级程序设计语言能够比脚本语言快10到20倍，这是由于高级程序设计语言不需要运行时的检查。

在接手一个特定的任务时，判断究竟是使用脚本语言还是高级程序设计语言可以考虑以下几个方面[4]：

·建立应用的主要工作是否是连接一些预先存在的主要构件？

·应用是否将处理多种不同种类的事物？

·应用是否包括 GUI？

·应用是否需要对许多串进行处理？

·应用的函数是否需要快速的研制？

·应用是否需要扩展？

对以上问题的肯定回答建议使用脚本语言。另一方面，对以下问题的肯定回答适合使用高级程序设计语言：

·应用是否执行复杂的算法和数据结构？

·应用是否需要处理大量的数据集，例如一幅图像中的所有像素，此时执行速度就变得较为关键？

·应用的函数是否定义明确且几乎不再改动？

脚本语言和高级程序设计语言是共生的。此两种语言的结合使用，能提供具有强大功能的编程环境。高级程序设计语言被用来编制供脚本语言使用的构件。例如，Javascript 中一个非常吸引人的特点是能够简单地使用 C 编写的 ActiveX 构件设计应用[5]。在 UNIX 中同样可以使用 C 编写的应用较容易地编写 shell 脚本。

四、脚本语言的应用

脚本语言已经存在很长时间，直到最近几年，多方面因素的结合才使其显得越发重要起来。其中最重要的因素是混合应用向粘贴应用的转换。这种转换的三个典型例子是 GUIs、Internet、构件框架。

1. 图形用户接口(GUI)

GUIs 首先出现于80年代早期，到了后期得到了广泛的应用。GUIs 目前在一个程序设计工程中占有全部工作的一半或更多。GUIs 是最基本的粘贴应用，其目的并不是建立一些新的功能而是建立起图形构件库和应用的内部功能之间的联系。

2. Internet

Internet 的广泛应用也促进了脚本语言的流行。Internet 从本质上来说就是一个粘贴工具，它并没有建立一些新的计算和数据，只是使已经存在的大量数据可方便地被访问。用于 Internet 程序设计任务的最理想的语言是能够很好地连接所有构件在一起的脚本语言，例如，Perl 被用来编写 CGI 脚本和 JavaScript 被用来编写 Web 页。

3. 构件框架

面向脚本应用的第三个例子是构件框架，例如 ActiveX 和 JavaBeans。高级程序设计语言适合于完成建立构件的工作，而脚本语言适合于组织构件成为应用的工作。如果没有一个好的脚本语言来处理构件，那么构件框架的强大功能就会失去。了解到这一点就会明白为什么构件框架在以 VB Script 为便利的脚本工具的 PCs 中取得了巨大的成功。

五、其它语言

这篇文章并不针对所有编程语言的全部特性。目前许多语言并不能明确地将其区分为高级程序设计语言或脚本语言，而是结合了此两类语言的各自的优点，例如，Visual Basic 语言就是这样一种混合型语言。它综合了脚本和高级程序设计语言各自的优点，既提供了大量 GUI 控件和对某些标准构件框架如 ActiveX 粘贴功能的支持，同时又可以运用属于高级程序设计语言类的标准 BASIC 语句和函数编写程序。目前许多传统的高级程序设计语言都在逐步发展为混合语言。就连以自由灵活、低级高效著称的 C 语言也是如此，如微软公司的 Visual C++6.0[6]。

参 考 文 献

1 Wall L, Christiansen T, Schwartz R. Programming Perl, 2nd ed 1996

2 Quigley E The UNIX Shell By Example. Copyright, 1997

3 Ousterbout J. Tcl and the Tk Toolkit, Addison-Wesley, Reading, Mass, 1994

4 Ousterbout J K. Scripting:Higher-Level Programming for the 21st Century. Computer, March 1998

5 Danesh A, Tatters W. JavaScript 1.1 Developer's Guide, 1998

6 Mueller J. Visual C++6.0从入门到精通，希望图书创作室译，1998,11

7 刘宗田. 程序设计方法学. 机械工业出版社出版，1992,10