物件導向語言OBJECT BASIC之設計與製作

高中組應用科學科第三名

台北市立建國高級中學

作　　者：劉　燈

指導教師：李　瑞

一、研究動機

近幾年流行的物件導向程式設計(OOP; Object-oriented Prog-ramming)標榜藉物件和「軟體IC」理念的實踐，降低軟體的複雜性並增加親和力。然而目前較廣泛被接受的物件導向語言，如C++、Object Pascal等，仍然需要「傳統的」語言訓練（如前者的C、後者的Pascal），對終端使用者(end-user)來說，想利用自撰程式達到一些獨特的需求，仍然相當困難。

在社團裡，和同學相互討論物件導向觀念之際，卻苦於缺乏合用的語言：Smalltalk不易取得；C++不易學習。這個研究便是在這樣的情況下開始的——希望利用上述的一些觀念和經驗，試著撰寫出一套物件導向的程式語言，讓物件導向觀念的討論不再是空中樓閣式的失望。這是它取名叫"Object Basic"(而不是BASIC)的原因。

二、研究目的

這個研究的主要目的，在於「程式語言」的設計與製作。希望達到的目標有：

㈠一個程式語言，包括完整的語法描述(language description)、編譯器(compiler)、執行環境(run-time environment)、相關工具等四項目。

㈡語言本身以物件導向程式設計的觀念作基礎，必需具備諸多操作物件的能力（見「研究過程與方法」一章）。

㈢語言本身儘量簡單易學，並製作足夠的相關工具。另外嘗試寫作一個輔助性的「整合操作環境」，便於使用者能即寫即用。

㈣數個供測試的程式，用以評量本研究結果之適用性。

三、研究設備器材

硬體部份：80486個人電腦一部。

軟體部份：MS-DOS version 5.00

Borland C++ compiler ver. 2.00

GNU RCS (Revision Control System) for MS-DOS

以上軟體部份，Borland C++是用以將Object Basic編譯器部份的原始程式碼編譯成可在機器上執行的機器碼；GNU RCS 則是所謂的「版本管理系統」，用以在研究過程中輔助程式碼的版本管理。

四、研究過程與方法

本研究的首要目的是設計製作一個新的「程式語言」。因此首先對個新的「語言」必需要有一個界定。[Friedman 1991]。什麼是物件導向程式語言？它具有那些特性？因此，研讀此方面相關資料及實際以現有的物件導向程式語言寫作數個程式，在研究過程中占了相當大的比重。

一般說來，一個物件導向程式語言的要素有以下幾個：

㈠物件與類別的關係：許多物件之間可能有一個共通的特性將之歸類便成「類別」(class)。

㈡物件的構成——狀態與行爲。

㈢訊息的傳送——訊息(message)是用來傳遞給一個物件，使物件觸動(trigger)一個方法(method)。

㈣繼承關係與抽像類別：「繼承」(inheritance)一詞的意義，與「母集合」、「子集合」的意義相同。

在了解物件導向的觀念後，再來才是語言的設計。研究過程中作者首先碰到的問題便是：一個「類別」的描述到底要不要屬於這語言的一部份？

像C++一類的語言，將「類別」的描述置入語言本身內，好處是程式設計者可一次修改操作，但便不能夠動態地(dynamically)增加或修改任何一個類別的描述。況且其編譯器的製作亦會較爲複雜。

Smalltalk-80語言採用互動式的「閱覽器」(browser)作爲物件操作和類別描述增修的工具。既然Object Basic要提供一個整合環境，那麼將類別描述增修排除在語言本身之外，在設計製作上會較爲方便。

經整理之後的語言要素如下：

㈠段落與句子。

㈡變數宣告。

㈢指派(assignment)。

㈣物件操作(object manipulation)：即訊息的傳送。

㈤流程控制(flow control)。

在下個階段，我們將要設計一個執行時期(run-time)的環境，用以管理及呈現(rep-

resent)眞實的物件。「物件機器」(object machine)便是此要求的構想。取名叫「物件機器」，因爲它與編譯程式裡的術語「虛擬機器」有神似之處〔Beck 1990〕。它負責執行由編譯器（下面將提到）所產生的程式碼，也動態地管理物件的生成、消滅、訊息傳送、類別繼承等關係。

整理歸納後，設計物件機器時應該考量的因素有：

㈠「物件」及「類別」的呈現。

㈡「方法」的掛入(hooking)。

㈢訊息的傳送與方法的執行。

㈣整合環境的配合。

最後一個項目便是編譯器的設計了。傳統上，編譯器的設計都分爲以下數個步驟：

㈠辭彙分析(lexical ana.)：分解最小單位。

㈡語法分析(syntactic ana.)：檢查有無文法上的錯誤。

㈢語意分析(semantic ana.)：了解一個「敘述句」(statement)的眞正意義。

㈣製碼。

一般說來以上四步驟又有單循環(single pass)和多循環(multi pass)之分別。後者便是將以上四步驟分開執行，前者則合併在一起一次完成。爲了設計方便，Object Basic採用的是前者，但也因此在文法的定義及編譯程式的流程上需特別小心的設計。

分析設計完之後便是實際的程式寫作了。這裡挑選了C程式語言來製作。原因是：一、C有良好的指標操作及動態記憶體管理；二、C所產生的碼相當精簡。並不是沒有考慮過以C++來實作，但C++已經是一物件導向語言，也已有其效率的瓶頸，加上其編譯器不如純粹的C一般普遍，因此只採用了C。

五、研究結果

最後完成的程式包括了一個Object Basic程式碼的編譯器，一個用以管理物件並執行編譯器所產生的程式碼的「物件機器」，並設計了一個命令列式的交談環境以方便使用者操作。

另外，作者也製作了一個「標準類別庫」(standard class hierarchy library)，包括了基本的資料結構，如整數、字串、抽像容器(abstract containers)、陣列、連結串列等。

配合圖形顯示，我們又另外以Object Basic製作一繪圖類別庫。如圓、矩形等圖元，均有抽像的「圖元」(Shape)之特性。以此建立一類別庫並加以測試。

六、討　　論

一個初具雛型的程式語言便以此架構完成。以語言的架構來說，Object Basic較接近Smalltalk、LISP等交談式操作的語言,尤其它需要一個「虛擬機器」的架構，使它的設計便不只是要考慮「編譯時期」的動作，也需要考慮「執行時期」(run-time)的種種狀況。

目前object Basic 較缺乏的便是錯誤的處理，也就是系統的強固性(stability)與可靠度(reliability)尚有賴加強。另外，它的執行時期中，有一大部份的時間在訊息的傳送和查詢比對(look-up)上，效率必不如傳統程式語言。

所幸，以物件導向語言的用途來論，互動操作及動態系統模擬這兩大用途本身就必需仰賴大量的訊息傳送和物件操作，因此這一點時間的犧牲，可以換得程式撰寫概念上的清晰。又CPU的速度加快，目前的硬體架構也已經有專爲物件導向系統設計的了（如NeXT公司的NeXTstep作業系統，即全部以Objective-C設計完成）。

七、結　　論

就應用(application)的觀點來說，Object Basic的語言要素少，相當便於學習。尤其物件導向的觀念，本就是我們習慣的思考方式之一。使用者不必費心於機器內部的動作，而可直接藉他人已建立的物件，將心思專注在創意上，就有點像牛頓所謂「站在巨人的肩膀上」了[陳建維1990]。而以C語言完成，也可以讓這套語言及其環境得輕易地從一種機器轉至另一種。而以它作應用程式的開發，或是軟體原型的製作(prototyping)，也或許是它的出路之一。

至此，最初的設計動機及構想終至初步地實現。

八、參考資料

陳建維．[1990]，從C到C++——物件導向革命，靖宇．

Aho, A.V. [1986], Compilers: principles, Technigues, and Tools. Addison-wesley.

Beck, Leland. [1990], System Software: An Introduction to System programming. Addison-Wesley.

Budd, Timothy. [1987], A Little Smalltalk.

Addison-Wesley.

Cox, Brad. [1987], Object Oriented Programming, An Evolutionary Approach.

Addison-Wesley.

Entsminger, Gary. [1991], The Tao of Objects. M & T Books.

Friedman, Linda. [1991], Comparative Programming Languages, Generalizing the Programming Function. Prentice-Hall.

Golderg, Adele and Robson, David. [1983]. Smalltalk-80:The Languag e and Its Implementation.Addison-Wesley.

Sethi, Ravi, [1989], Programming Languages,Concepts and Constructs. Addison-Wesley.

Wirth, Niklaus. [1986], Algorithms & Data Structures.Prentice-Hall.

評語

1. 物體導向爲軟體發展趨勢，作者以基本程式技巧，表現物件導向觀念。
2. 以高中生程度，能了解並發揮此觀念，殊屬不易。
3. 使用的是基本Compiler技巧，雖然有更方便之軟體發展工具（如Yacc），以高中生之計算機知識，亦爲正確之作法。
4. 題目似可稍做修改。Object Basic易與Turbo Quick Basic聯想，以爲是在Basic語言上發展的Object Language。
5. 使用的例子爲任一物件導向之共同例子,以畫不同形狀的圖來表達00觀念。建議換別個例子,以表現融會貫通，充分了解，舉一反三的實力。