# 整合變易理論之物件導向程式設計語法導正系統

黃子峰　　蘇建元　　張証欽　　王宗一*
國立成功大學工程科學所
n96001177@mail.ncku.edu.tw, bredysu@gmail.com, johnc680106@gmail.com,
wti535@mail.ncku.edu.tw*

## 摘要

物件導向程式設計是資訊軟體專業領域中十分受到重視的課程，但初學者往往在程式撰寫的過程中，對程式語法、語意或指令敘述的程式設計概念一知半解而在真正面對程式實際撰寫的過程中感到挫折，甚至害怕接觸程式。本研究提出整合變易理論策略並建構出用以輔助與導正初學者物件導向程式語法的學習系統。以變異理論其讓學習者體驗與分辨特徵及差異的學習方式下，讓學習者在撰寫 C++物件導向程式、編譯程式碼的過程中，給予學習者更有效的語法導正回饋訊息，以幫助他們學習。

為了探討變易理論應用於物件導向程式設計輔助語法導正的學習成效，本研究以實驗設計方式進行，共有 90 位大一新生參與實驗，以 S 型分組方式將參與學生分派為「具變易理論策略組」(實驗組）與「無變易理論策略組」(對照組)，以單因子變異數分析進行統計分析；並結合問卷分析以及訪談，探討本研究的輔助語法導正機制對於學習者的影響。實驗結果顯示使用「具變易理論模式的語法導正策略」在學習成效表現上優於使用「無變易理論模式的語法導正策略」，證明應用變易理論於物件導向程式設計的語法導正學習能夠幫助學習者撰寫及練習物件導向程式，並且能解決在學習程式設計的過程中，對程式語法、語意或指令敘述的不了解而感到挫折的問題。

**關鍵詞**：變易理論；物件導向程式設計；語法導正輔助

## 1. 前言

程式設計一向是資訊技術人員不可或缺的技能之一，要能應用、撰寫出合用的程式，是需要不斷的自我練習、撰寫以及除錯，許多人寫不好程式，或害怕接觸程式，在於對程式的基本概念不瞭解或有所混淆，以致於學習感到挫折[1]-[4]，甚至逃避或放棄學習[5]。因此，本研究採用變易理論（Variation Theory）方法，於學生在編寫程式設計的過程中，以特殊的類比、對比、區別、融合四種不同模式的學習輔助，提供針對語法錯誤以及錯誤程式概念的矯正，來幫助學習者釐清或分辨程式概念間的差異，達成矯正錯誤程式觀念的目的。

## 2. 相關文獻

### 2.1 程式設計教學策略

為了有效促進程式設計的學習，過去一些有效的教學策略被廣泛應用在程式設計的教學中，例如使用類比教學方法於程式設計教學[6]；或透過動態視覺化程式碼的呈現方式來幫助學習者學習程式設計[7]；於遞迴程式設計教學中運用雙碼理論建構概念模型的學習策略等[8]。

雖然這些策略皆有助於程式語言的學習，但有文獻指出學生對程式設計的撰寫上仍深感困難[9][10]，推敲其原因在於對程式概念及程式語法不熟悉或產生誤解。針對該問題，有學者認為應透過許多淺顯易懂的例子來幫助學習者釐清程式概念中容易混淆的部分[11]。另也有學者認為在程式教學中，應針對易混淆的概念善用區別、對比等方式以突顯出概念的關鍵屬性，讓學習者更有效率、有效的掌握概念的核心意涵[4]。

### 2.2 變易理論教學模式

源自於現象圖析學（phenomenongraphy）的變易理論（variation theory），強調體驗事物的變化和差異才是認知的開始，透過分辨現象或概念的特徵及差異，才能習得事物或概念的本質[12]。並將整個變易理論細分成四種不同的變易模式[13]，由淺入深分別為「類比」、「對比」、「區別」以及「融合」，分述如下：

1. 類比（generalization）：要了解某事物的普遍性，就必須經歷此事物在不同情況下的呈現。例如教某個概念時，透過列舉不同的例子，讓學習者辨識例子之中相同的地方，以認識此概念。
2. 對比（contrast）：要清楚瞭解某事物，就必須把此事物和其他事物進行對比。更具體而言，若是你想知道什麼是 X，就必須知道什麼不是 X，例

如要認識「甜味」，就要經歷苦味及鹹味，才能知道什麼「是甜味」，什麼「不是甜味」。

3. 區別（separation）：要審辨事物某方面的特徵，就必須經歷此事物在某方面的改變，而保持其他方面不變，如讓學習者思考同一問題的不同答案或同一現象的不同解釋。
4. 融合（fusion）：某事物需要我們綜合考慮各方面因素時，我們必須經歷多個關鍵性特徵的改變，同時察覺到各方面的轉變。

該策略曾廣泛應用於各個學科領域來幫助學生釐清迷思概念，其中亦包含程式設計的教學。例如 Suhonen 等人在教學時就運用變易理論來突顯程式設計概念差異[14]。Eckerdal 利用變易理論讓學習者觀察二段相似的程式碼，讓學習者透過對比與區別程式碼及比對執行結果的差異來幫助學生習得正確概念[15]。Thota 等人運用變易理論於 OOP 課堂教學，協助學習者學習 OO 概念[16]。

## 3. 研究方法

本研究針對初學者建構一物件導向程式設計的概念與語法導正學習系統，以變易理論的類比、對比、區別及融合為基礎，該系統主要涵蓋兩個層級，分別為語法導正機制與變異理論回饋策略。

### 3.1 語法導正機制

本研究採行的語法導正機制流程如圖 1 所示，在系統中讓學習者實際進行程式碼的撰寫，當學習者輸入的程式碼經編譯器剖析之後發生編譯錯誤，將啟動語法導正機制進行語法導正，而在學習者反覆修正及編譯程式碼的過程中，語法導正機制會依照變易理論的教學模式，逐步地給予學習者適當的回饋訊息。

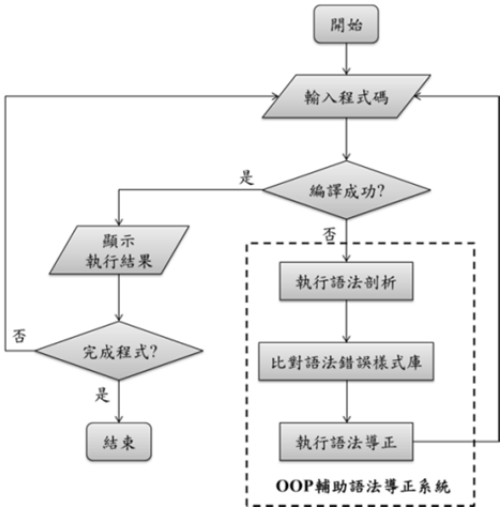


**圖 1 初步擬定的語法導正機制流程圖**

本研究所建置之語法導正方法主要參考自[17]針對協助學習者程式除錯，所提出的虛擬助教機制（Virtual Teaching Assistant，VTA）。本研究經適度調整該 VTA 機制並用以結合變易理論，並使用語法錯誤樣式庫的錯誤樣式內容來啟動語法導正知識庫所對應之導正教學。其修改後的方法如表 2 所示。

**表 2 修改後的 VTA 機制**

| 提示層級 | 提示方法 |
|---|---|
| hint 1 | 告訴學習者程式中「哪裡有錯」與「程式錯誤類型」。 |
| hint 2 | 針對錯誤，提供相關的語法知識或語法導正教材，即**啟動「變易理論語法導正模組」**。 |

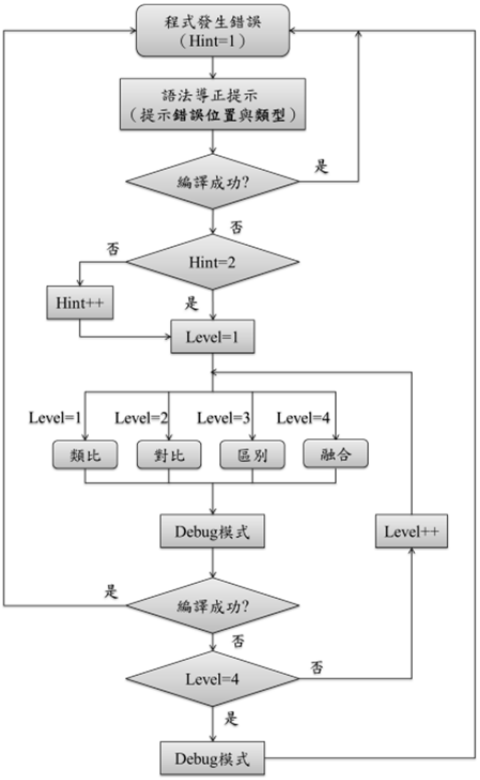


**圖 2 變易理論的語法導正流程圖**

所新設計之詳細語法導正流程如圖 2 所示。當程式發生編譯錯誤時，給予學習者 hint 1 的提示，告訴學習者程式「錯誤的位置」以及「程式錯誤類型」；若學習者仍無法完成修正語法，接著將依照變易理論模式，給予類比模式（Level=1）的回饋教材，並回到 debug 模式讓學習者修正程式，若糾正錯誤後仍發生同樣錯誤修正程式，若糾正錯誤後仍發生同樣錯誤情形，則系統進一步給予對比模式（Level=2）的回饋訊息，當學習者一再地遇到同樣的程式錯誤情形，系統將依序以類比、對比、區別、融合等四個模式的回饋，逐一予以提示。

### 3.2 變易理論的回饋策略

在此以物件導向程式概念中的私有區存取錯誤案例為例（如圖 3），該程式錯誤乃是因為物件呼叫到私有區的成員函式所導致，在 C++的語法定義中，如果某類別內的成員被定義在私有區（private），則這些成員僅能透過類別內的成員函式對其進行存取。針對發生此錯誤情形，則設定依照特定狀況提供變易理論四種模式的語法導正教材內容，說明如下：

```
1  #include <iostream>
2  using namespace std;
3  class Circle
4  {
5  private:
6     double radius;
7     double Area();
8  };
9
10 double Circle::Area()
11 {
12    return radius * radius * 3.14;
13 }
14
15 int main()
16 {
17    Circle C1;
18    cout << "Area of Circle C1 is " << C1.Area();
19
20 return 0;
21 }
```

成員函式「Area()」為私有區的成員

**圖 3 私有區存取錯誤實例**

**表 3 類比模式的範例**

(1)

```
#include <iostream>
using namespace std;
class MyClass
{
private:
   void display();   // private member function
};

void MyClass::display()
{
    cout << "print from the member of MyClass";
}

int main()
{
   MyClass obj;
   obj.display();  // Call a private member function, Error!

return 0;
}
```

(2)

```
#include <iostream>
using namespace std;
class MyClass
{
private:
   double data;     // private data member
};

int main()
{
   MyClass obj;
   obj.data = 1.1;  // Access the private data member, Error!

return 0;
}
```

(3)

```
#include <iostream>
using namespace std;
class MyClass
{
   void display();   // 若未指定成員之權限，則預設為 private
};

void MyClass::display()
{
   cout << "print from the member of MyClass";
}

int main()
{
   MyClass obj;
   obj.display();    // Error!
return 0;
}
```

■ **Level 1 – 類比模式**

變易理論強調要讓學習者了解某事物的普遍性，就必須讓其學習者經歷此事物在不同情況下的過程體認。因此以私有區存取錯誤實例（如圖 3）為例，透過觀視不同的例子，讓學習者辨識不同例子之中相同的地方來認識此概念，藉此反思所撰寫的程式發生的錯誤在哪，該回饋資訊如表 3。

| | |
|---|---|
| | |
| | |

■ **Level 2 – 對比模式**

對比模式主要強調要讓學習者清楚了解某事物，就必須把此事物和其他事物進行對比，藉以突顯某事物的特徵。如表 4，便是讓學習者透過經歷「私有區成員」與「公有區成員」兩者關鍵特徵的變化與異同，以幫助學習者瞭解程式為什麼發生錯誤。

**表 4 對比模式的範例**

1. 「private 成員」大多為**資料成員**，**只能**透過**類別本身**的成員函式之程式命令對其做存取，其他外部程式命令皆**不允許透過「.」直接存取私有成員**。
2. 「public 成員」大多為**成員函式**，**除了**可透過**類別本身**的成員做存取之外，公開的成員**可任意為外部程式命令透過「.」直接存取**。

■ **Level 3 – 區別模式**

要釐清事物某方面的特徵，就必須經歷此事物在某方面的改變，而同時保持其他方面不變，如讓學習者思考同一問題的不同答案或同一現象的不同解釋。因此針對該私有區存取錯誤的實例（如圖 3），區別模式所給予的回饋訊息如表 5 所示，系統提供學習者 2~3 個此錯誤類型的正確程式範例，亦即讓學生經歷「程式」某方面改變（由錯誤改為正確），而「錯誤類型保持不變」，讓學生能比較自己的錯誤程式與正確範例差別後，瞭解該部分正確的程式應如何撰寫，以導正程式錯誤。

**表 5 區別模式的範例**

(1)

```
#include <iostream>
using namespace std;
class MyClass
{
public:
   void display();   // public member function
};

void MyClass::display()
{
    cout << "print from the member of MyClass";
}

int main()
{
   MyClass obj;
   obj.display();    // Correct!

return 0;
}
```

(2)

```
#include <iostream>
using namespace std;
class MyClass
{
public:
   double data;     // public data member
};

int main()
{
   MyClass obj;
   obj.data = 1.1;    // Correct!

return 0;
}
```

■ **Level 4 – 融合模式**

某事物需要綜合考慮各方面因素時，我們必須經歷多個關鍵性特徵的改變，以察覺其各項轉變，

如讓學生透過同一現象的不同感知，以開啟他們思辨和學習的空間，來習得相關知識或概念。因此，融合模式將如表 6，系統將以一個同時包含「正確範例」與「錯誤情形」的程式碼作為該階段的回饋，當學習者經歷過前面三個階段，已將事物的學習特徵逐一區別出來之後，再將這些特徵於同一情境當中融合表示，可以提升學習者對於程式語法的理解，更能全面、完整的掌握正確的程式語法。

**表 6 融合模式的範例**

```
#include <iostream>
using namespace std;
class MyClass
{
private:
   void testing();     // private member function
public:
   void display();     // public member function
};

int main()
{
   MyClass obj;
   obj.testing();     // Error!

   obj.display();     // Correct!

return 0;
}
```

### 3.3 系統架構

本研究所建置的系統共包含六個部分，其架構如圖 4 所示。

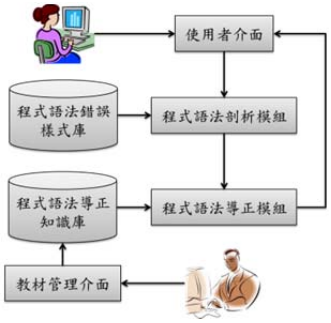


**圖 4 系統架構圖**

學習者可以透過「使用者介面」進行 C++物件導向程式的撰寫、編譯及除錯，並在程式發生編譯錯誤時，系統將予以程式語法導正幫助學習者進行除錯。

「程式語法剖析模組」主要功能為剖析、分析學習者輸入的程式碼，透過比對「程式語法錯誤樣式庫」以得知該程式的錯誤類型。「語法錯誤樣式庫」則存放學習者於程式撰寫時常犯的語法錯誤樣式。而「程式語法導正模組」則協助學習者進行除錯，將語法剖析模組剖析後之結果結合「程式語法導正知識庫」內的變異理論模式回饋資訊進行語法導正。其可能發生的所有錯誤樣式處理規則，本研究聘請三位具 OOP 教學經驗的專家教師，於「教材管理介面」中以變易理論之類比、對比、區別與融合等模式為原則製作相對應的變易理論語法導正回饋教材。

## 4. 實驗設計

為驗證所開發出的「整合變易理論之 OOP 語法導正學習系統」的效果，實驗設計將以期中考試成績透過 S 型分派學生為實驗組和控制組，使其兩組學生具有相同的程式設計先備知識。自變項為語法導正策略，依變項為學習成就測驗結果。其中，實驗組使用「具變易理論模式的語法導正策略」的導正教材；控制組則使用「無變易理論模式的語法導正策略」的教材，而控制組修改後的 VTA 機制如表 7 所示，在提示層級 1 中同樣告訴學習者程式之「錯誤位置」及「錯誤類型」；而在層級 2 中則使用原始編譯器資訊做為回饋。

根據課程進度，我們將測驗的範圍訂定為「物件與類別」及「繼承」兩單元，讓學生使用本系統實際進行 OOP 程式的撰寫，並依測驗範圍分為兩次實驗活動，將在實驗 1（物件與類別單元的上機測驗）之後間隔一個月，執行實驗 2（繼承單元的上機測驗）。每次各提供 2 題程式問題，學生須於規定時間（120 分鐘）內完成所要求的程式撰寫。學生完成、提交後的結果將由人工進行批改，做為學習成就測驗的結果。

**表 7 控制組之修改後的 VTA 機制**

| 提示層級 | 提示方法 |
|---|---|
| hint 1 | 告訴學習者程式中「哪裡有錯」與「程式錯誤類型」。 |
| hint 2 | 提供原始編譯器回饋資訊（使用 Microsoft Visual C++編譯器）。 |

### 4.1 實驗對象

實驗對象是以台灣南部某大學工程科學系之必修程式設計課的 90 位修課學生，分為實驗組與控制組各 45 人，兩組均同時接受過程式設計課程講授。我們利用實驗分析以及問卷、訪談方式來檢驗變異理論於物件導向程式設計語法導正系統中的效果。相關分析結果如下:

### 4.2 統計分析結果

扣除無效樣本得第一次實驗活動的有效樣本數為 82 人，第二次實驗活動則為 81 人。本研究採單因子變異數分析（ANOVA），進行變異數差異的顯著性考驗。於正式實驗前先執行 Levene’s 變異數同質性檢定來初步評估實驗組與控制組是否具同質性（如表 8），其結果得知二組沒有顯著差異（$p = .891$ 與 $p = .976$），顯示兩組學生具同質性。

**表 8 先備知識變異數同質性檢定**

| | *df1* | *df2* | *Sig.* |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **實驗 1** | 1 | 80 | .891 |
| **實驗 2** | 1 | 79 | .976 |

其針對學習成效測驗進行單因子變異數分析，結果如表 9 所示，第一次實驗分析結果為 F(1,80) = 4.255，p = 0.042；第二次實驗結果為 F (1,79) = 4.165，p = 0.045，兩次不同單元的實驗均達顯著差異。可推論出實驗組學生使用具變易理論模式的 OOP 語法導正的學習系統，比無學習變易理論模式的 OOP 語法導正策略的學習系統的學生，在學習成效上有較好的表現。

**表 9 學習成效之變異數分析**

| | | | | | | |
|---|---|---|---|---|---|---|
| ***實驗1*** | 組間 | 4842.375 | 1 | 4842.375 | 4.255 | 0.042* |
| | 組內 | 91054.076 | 80 | 1138.176 | | |
| | 總和 | 95896.451 | 81 | | | |
| ***實驗2*** | 組間 | 4240.400 | 1 | 4240.400 | 4.165 | 0.045* |
| | 組內 | 80420.879 | 79 | 1017.986 | | |
| | 總和 | 84661.279 | 80 | | | |

*$p < 0.05$

為進一步了解具變易理論模式的語法導正策略對不同程度學生的影響，我們進一步從收集到的數據資料中將參與實驗的學生依照先備知識成就測驗結果（學期第一次期中考試成績），分成高、中、低成就三群，並從兩次實驗的成就測驗分數觀察發現，本研究所建置之整合變易理論的 OOP 語法導正系統對於中成就的學生幫助較大，對於低成就學生次之，而對於高成就的學生則較小。

## 4.3 問卷結果分析

為深入了解變異理論輔助語法導正機制對於學習者的使用意見，由三位熟悉物件導向程式設計教學的專家教師共同編制「變易理論語法導正學習系統使用問卷」，所有試題以李克特五點量表分成非常同意、同意、無意見、不同意與非常不同意五個選項，其問卷具備一定程度之專家效度，該問卷針對實驗組的學習者進行調查，且僅針對問卷進行描述性分析，用以了解實驗組學習者於使用該語法導正系統後，對於該應用變易理論於語法導正回饋的看法與意見，問卷內容共分為「類比模式」、「對比模式」、「區別模式」、「融合模式」及「整體經驗」等五個部分進行調查。其問卷統計結果如表 10 所示，對於「整體經驗」向度中的分析結果為贊同（平均數=3.65、3.50、3.78），而對於變易理論中四種變易模式的語法導正回饋也大多表示贊同，其中大部分填答者較贊成「對比模式」的回饋有幫助（平均數=4.03、3.75），而其次為「區別模式」（平均數=3.85、3.65）及「融合模式」（平均數=3.70、3.63），最末為「類比模式」（平均數=3.58、3.40、3.45）。

**表 10 問卷之平均數與標準差統計表**

| | 題目 | 平均數 | 標準差 |
|---|---|---|---|
| **類比模式** | 1 有助於辨識程式錯誤的地方。 | 3.58 | .68 |
| | 2 有助於清楚釐清相關程式語法的特徵。 | 3.40 | .71 |
| | 3 有助於未來遇到相同語法錯誤時，辨識相同的錯誤類型。 | 3.45 | .75 |
| **對比模式** | 4 有助於瞭解程式「為什麼發生錯誤」。 | 4.03 | .48 |
| | 5 有助於清楚釐清相關程式語法的特徵。 | 3.75 | .59 |
| **區別模式** | 6 有助於瞭解正確的程式語法。 | 3.85 | .66 |
| | 7 有助於分析程式錯誤的地方。 | 3.65 | .80 |
| **融合模式** | 8 有助於更全面、更完整地理解程式錯誤，並能進一步修正該錯誤。 | 3.70 | .65 |
| | 9 有助於更全面地掌握正確的程式語法。 | 3.63 | .74 |
| **整體經驗** | 10 有助於釐清以往程式語言語法上的模糊觀念，得到正確的語法概念與認知。 | 3.65 | .62 |
| | 11 以「逐層、漸進」方式提供回饋有助於對程式語言語法觀念進行更深入思考。 | 3.50 | .72 |
| | 12 整體而言有助於「導正」程式語法錯誤。 | 3.78 | .66 |

## 4.4 訪談結果

從實驗組學生的訪談結果可以發現到，學生樂於透過該系統進行程式撰寫，主要能針對編譯錯誤的部分，得到更完整的回饋訊息。該組學生也反應 hint 1 回饋提示的「錯誤位置」與「程式錯誤類型」，在已經領悟到是屬於哪種錯誤情形或糾錯方式時，相較於一般的程式編譯方式，該系統所提供的回饋資訊將更有助於他們糾正程式錯誤。其中，學生表示所系統所提供的「類比」回饋訊息能幫助思考與判斷較簡單的程式錯誤，當程式出錯的內容較為複雜或難解時，「區別」回饋則能幫助思考與反思較複雜、困難的程式問題。「對比」回饋主要是提供錯誤的關鍵特徵，讓學生進行比較，不少學生皆反應該方式對他們的學習助益最大；「融合」回饋亦能提供完整的例子，幫助學生解決錯誤與提供詳細的程式錯誤論述訊息。

在控制組的學生訪談中，一些學生認為原始編譯器的回饋資訊較不容易讓人理解；而尤其當所撰寫的程式發生多種錯誤情形時，原始編譯器回饋資訊可能過度複雜，而造成學習者無法糾正他們的程式錯誤；也有學生表示若回饋資訊能提供更好的程式例子來參考，將會有很好的幫助。

整體而言，整合變異理論的四種模式於物件導向程式設計語法導正學習中，對於學生語法的釐清

及糾正程式錯誤有正向的助益。

## 5. 結論與建議

本研究開發整合變易理論於物件導向程式語言語法導正學習系統，為了檢驗該變異理論應用於語法導正學習系統之成效，讓已學過程式設計的初學者進行實際的程式撰寫，並以兩階段的上機進行程式撰寫實驗進行，結果顯示使用「具變易理論模式的語法導正策略」的學生於學習成效表現上優於「無變易理論模式的語法導正策略」的學生，推論出變易理論應用於物件導向程式設計語法導正能有益於學習者釐清程式語法、語意或指令敘述的錯誤概念。未來將持續延伸至其他物件導向程式設計單元，或針對不同程度的學習對象，進一步印證與深入探討該變異理論策略中不同模式與不同學習單元對學習者的影響。另外，對於系統的使用建議，亦可強化系統的自我操作性，如提供程式編輯區中關鍵字顏色分別、自動對齊、自動縮排等等的編排功能，以提供更完整、妥善的物件導向程式設計語法導正學習系統。

## 致謝

本研究承蒙台灣國科會經費之補助，計畫編號為 NSC 101-2511-S-006-011-MY2，特此致謝。

## 參考文獻

[1] G. Stavroula and S. R., "Using Educational Tools for Teaching Object Oriented Design and Programming.," *Journal of Information Technology Impact*, vol. 7, pp. 111-130, 2007.

[2] A. Ebrahimi and C. Schweikert, "Empirical study of novice programming with plans and objects," *SIGCSE Bull*, vol. 38, pp. 52-54, 2006.

[3] A. Eckerdal and M. Thun'e, "Novice Java programmers' conceptions of "object" and "class", and variation theory.," *Proceedings of the 10th Annual SIGCSE Conference on Innova-tion and Technology in Computer Science Education (ITiCSE'05)*, pp. 89-93, 2005.

[4] S. Holland, R. Griffiths, and M. Woodmanw, "Avoiding object misconceptions.," *In Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education(SIGCSE'97)*, pp. 131-134, 1997.

[5] 陳怜秀、郭英峰, 學生程式設計能力影響因素之研究—以崑山技術學院資訊管理系為例, *技術學刊*, vol. 13, pp. 661-668, 1998.

[6] J. Whittle, A. Bundy, and H. Lowe, "Supporting programming by analogy in the learning of functional programming languages.," presented at the The 8th International Conference on AI in Education (AIED), 1997.

[7] H. Ramadhan, "An intelligent discovery programming system.," in *SAC '92 Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing: technological challenges of the 1990's*, New York, USA, 1992, pp. 149-159

[8] 陳明溥, 雙碼理論於遞迴程式設計教學之概念模型設計研究, 第八屆電腦輔助教學國際研討會, 逢甲大學, 1999.

[9] J. Bennedsen and M. E. Caspersen, "Revealing the Programming Process," *Proceedings of 36th SIGCSE Technical Symposium on Computer Science Education (SIGCSE2005)*, pp. 186-190, 2005.

[10] V. Shanmugasundaram, P. Juell, G. Groesbeck, and M. Makosky, "Evaluation of Alice World as an introductory programming language.," in *Proceedings of the ED-MEDIA 2006-World Conference on Educational Multimedia, Hypermedia & Telecommunications*, 2006, pp. 1976-1982.

[11] 陳明溥, 程式語言課程之教學模式與學習工具對初學者學習成效與學習態度之影響, *師大學報*, vol. 52, pp. 1-21, 2007.

[12] F. Marton, "Phenomenography-describing conceptions of the world around us.," *Instructional Science*, pp. 177-200, 1981.

[13] 祁永華, 謝錫金, and 岑紹基, *變易理論與學習空間*. 香港: 香港大學出版社, 2005.

[14] J. Suhonen, E. Thompson, J. Davies, and Kinshuk, "Applications of variation theory in computing education," in *Proceeding Koli Calling '07 Proceedings of the Seventh Baltic Sea Conference on Computing Education Research*, Koli National Park, Finland, 2007, pp. 217-220.

[15] A. Eckerdal and M. Thun'e, "Variation Theory Applied to Students Conceptions of Computer Programming.," *European Journal of Engineering Education*, vol. 34, pp. 339-347, 2009.

[16] N. Thota and R. Whitfield, "Holistic Approach to Learning and Teaching Introductory Object-Oriented Programming.," *Computer Science Education*, vol. 20, pp. 103-127, 2010.

[17] C.-Y. Chou, B. Huang, H., and C. J. Lin, "Complementary Machine Intelligence and Human Intelligence in Virtual Teaching Assistant for Tutoring Program Tracing," *Computers & Education*, vol. 57, pp. 2303-2312, 2011.