

## ezHandle

專題編號：106-CSIE-S018

執行期限：105年第1學期至106年第1學期

指導教授：鄭有進

專題參與人員：103820017 馮聖昌

### 一、摘要

本專題主要探討軟體例外處理行為之塑模。目前國內外有關軟體例外處理行為，其程式碼大多是由程式設計師根據循序圖及自身的經驗所撰寫。然而，若軟體開發時需要使用第三方之應用程式介面，程式設計師則需閱讀相關文件使能了解其例外處理行為再加以撰寫程式。本專題以剖面導向程式設計取得第三方軟體之模型，再以自動產生之測試作機器學習之訓練，使其能夠正確表現其第三方軟體之例外處理行為，進而程式設計師更有效率地得知軟體例外處理之行為。

**關鍵詞：**例外處理、應用程式介面、剖面導向程式設計。

### 二、緣由與目的

目前普遍有關軟體例外處理行為，其程式碼大多是由程式設計師根據循序圖及自身的經驗所撰寫。然而，若軟體在其開發時需要使用第三方之應用程式介面，程式設計師則需閱讀相關文件使能了解其例外處理行為後再加以撰寫程式。再者，若第三方之應用程式介面並無釋出相關文件，則例外處理程式的撰寫將會有一定的難度，增加軟體開發所需之時數。而本專題使程式設計師無需閱讀相關文件便能得知其例外處理行為，進而減少軟體開發所需之時數。

### 三、研究技術與方法

#### (一) AspectJ

一種 Java 剖面導向程式設計的介面，用於外加程式碼以改變程式行為而不更動原程式碼。

#### (二) And-or graph

用於表達解題技巧及程式運作方式之方法。本專題用於表達例外處理行為。

#### (三) Machine learning

以訓練集合歸納出來之假設來預測未來發生情形之技術。

### 四、使用工具說明

#### (一) Eclipse

可作為 Java 應用程式與 Android App 之開發環境。

#### (二) SWI-Prolog

可作為 Prolog 應用程式之開發環境。

#### (三) Atom

可作為程式碼編輯之環境。

### 五、架構流程

#### (一) 建立初步模型

以 AspectJ 取得該程式正常之模型，並以 and-or graph 表示其關係。

#### (二) 根據模型產生測試

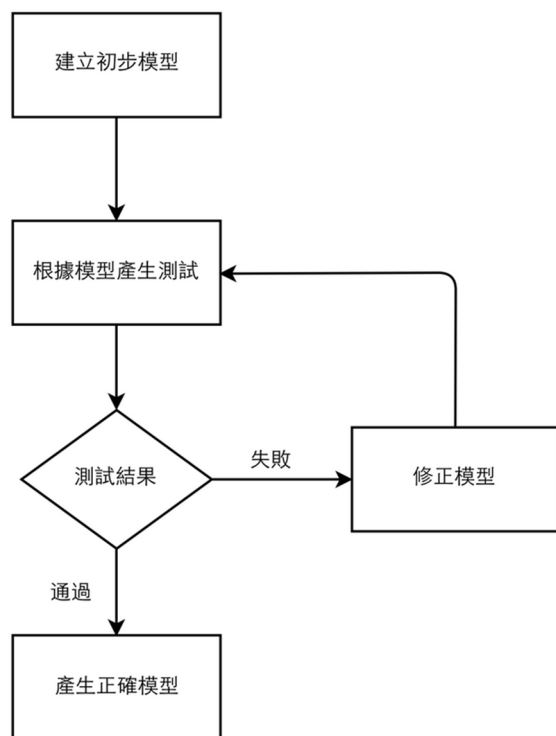
透過樹的尋訪，產生對應的例外測試。

### (三) 依測試結果決定是否修正模型

若全部測試皆通過，則該模型為其例外處理之行為；若有失敗之測試，則需修正原先之模型。

### (四) 修正模型

以測試實際尋訪之方法預測其正確例外處理之行為並加以修正。

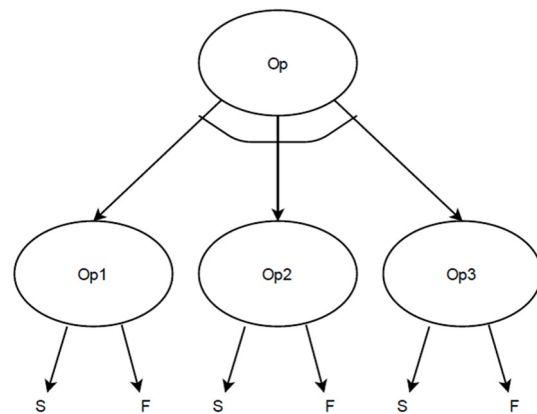


圖一、架構流程圖

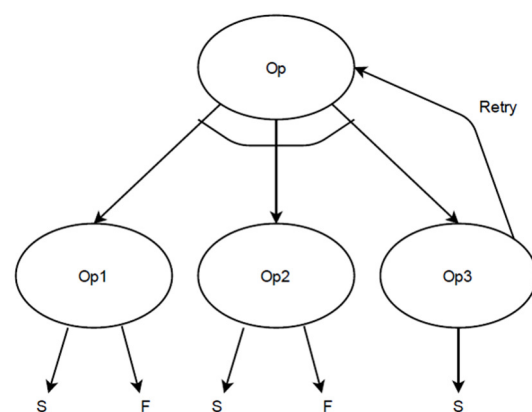
## 六、專題成果與結論

在不斷的修正之後，做出了這個程式的初型，目前可以判斷三種例外處理行為，但在 retry 的部分還尚未能判斷其重複之次數。

經過這次的實務專題研究，一個人實作可以學到許多東西，看似這是一件好事，可以避免掉爭執或是整合的問題，但是不能像其他組一樣有著多方的意見，有時候在解決問題時，花的時間會比其他組來得多，不過還是獲益良多！



圖二、程式之初步模型



圖三、程式修正後之模型

## 七、參考文獻

[1] AspectJ tutorial

<https://eclipse.org/aspectj/doc/next/progguide/starting.html>

[2] Prolog

Ivan Bratko, Prolog Programming for Artificial intelligence 4<sup>th</sup> Edition

[2] Prolog Question

<http://www.swi-prolog.org/FAQ/>