



**國立臺北科技大學**

**資訊工程系碩士班**

**碩士學位論文**

**支援多國語言的 Robot Framework 網頁自動  
化驗收測試工具的功能改善與擴充**

**Further Improvement and Extension of an  
Automated Web Testing Tool  
with Robot Framework to Support  
Internationalization**

**研究生：林稟宸**

**指導教授：鄭有進、謝金雲**

**中華民國 110 年 6 月**

# 摘要

論文名稱：支援多國語言的 Robot Framework 網頁自動化驗收測試工具的功能改善與擴充

頁數：68 頁

校所別：國立台北科技大學 資訊工程系碩士班

畢業時間：一百零九學年度第二學期

學位：碩士

研究生：林稟宸

指導教授：謝金雲、鄭有進教授



相關名詞：Robot Framework、XPath、關鍵字 (Keyword)、代理關鍵字 (Keyword Proxy)、i18n

在現今的國際化社會，一個網頁可能同時擁有多種語言的版本。根據朱峻平的論文「支援多國語言的 Robot Framework 網頁自動化驗收測試」所提出的第一版 i18n 工具，目前僅能支援七種 Robot Framework 原生關鍵字的翻譯代理，且也只能對三種 XPath 內的 HTML 屬性進行翻譯；本論文將針對以上兩點，對第一版 i18n 工具進行代理關鍵字的擴充和 XPath 翻譯邏輯的改善。除此之外，並會提供一詞多譯的圖形化使用者介面讓使用者去選擇期望的翻譯，以解決一詞多譯的問題。最後，將 i18n 工具包裝成為可以透過 pip 安裝的 Python 模組。

# ABSTRACT

Title: Further Improvement and Extension of an Automated Web Testing Tool with Robot Framework to Support Internationalization

Pages: 68

School: National Taipei University of Technology

Department: Computer Science and Information Engineering

Time: June, 2021

Degree: Master

Researcher: Bing-Chen Lin

Advisor: C.Y. Hsieh Ph.D. and Y.C. Cheng Ph.D.



Related terms: Robot Framework, XPath, Keyword, Keyword Proxy, i18n

Nowadays a web page may have versions in multiple languages. This thesis extends the thesis “Internationalization Support for Automated Web Testing with Robot Framework” by Chun-Ping Chu, which supports only seven Robot Framework native keywords and is limited to three HTML attributes as translation targets. In addition to supporting more keywords and HTML attributes, the extension in this thesis also supports user’s choice of translations of words and installation by pip.

# 誌謝

首先感謝父母對我從小到大的栽培，也感謝教授在北科的這兩年，不但教我們程式設計，也讓我從實驗室不輕鬆卻充實的日常生活中收穫良多。也感謝實驗室的同學們、學弟妹兩年的陪伴。

在論文的研究過程中，感謝發哥提供了我許多寶貴的意見與討論，也特別感謝朱峻平學長，工作之餘抽空指引我論文的研究方向，共同討論如何讓此 i18n 工具成為一個更好的設計。

不期望可以做出多偉大的軟體，但要做出一個有用的軟體。本論文秉持著這一精神，盡力完善，不好或缺失的地方也在論文限制處如實敘述，希望透過現在的努力與之後的維護，能讓「在不改動腳本下，實現多國語言網頁自動化驗收測試」的 i18n 工具，在日後成為測試者們的一個選項。

最後，也和自己說聲辛苦了，寫論文不像平時隨手寫一些有趣的專案那般信手拈來，要說過程中完全沒有遭遇挫折，我相信那是騙人的。謝謝自己的堅持，才有了今天的成果，也祝自己早日交到女朋友。

程式雖然不像人一般有感情，但一旦投入心血，它卻不會背叛你，共勉之。

# 目錄

中文摘要	i
ABSTRACT	ii
誌謝	iii
目錄	iv
表目錄	vii
圖目錄	viii
第一章 緒論	1
1.1 研究背景與動機	1
1.2 研究目標	2
1.3 論文組織架構	3
第二章 背景知識	4
2.1 i18n 工具相關知識	4
2.1.1 國際化	4
2.1.2 自動化驗收測試	4
2.1.3 XPath	5
2.1.4 JSON 格式翻譯檔在前端框架中的運作	5
2.2 Robot Framework	7
2.2.1 Robot Framework 關鍵字	7
2.2.2 Robot Framework 測試腳本	8

2.2.3	Robot Framework 測試報表	8
2.2.4	Robot Framework Listener	8
2.3	第一版 i18n 的系統介紹	9
2.3.1	第一版 i18n 的系統架構	9
2.3.2	第一版 i18n 的系統執行流程	11
2.3.3	產生一詞多譯問題的緣由	15
第三章	新版 i18n 的系統設計	16
3.1	系統擴充	16
3.2	擴充與修改代理關鍵字實作	19
3.2.1	新版 i18n 需要擴充的代理關鍵字	19
3.2.2	未來遭遇 Robot Framework 相關 Library 版本更新的應對	22
3.2.3	新版 i18n 擴充代理關鍵字的實作步驟	23
3.3	改善 XPath 翻譯邏輯使其能應對各種 HTML 屬性	27
3.4	提供圖形化使用者介面解決一詞多譯的問題	30
3.5	將 i18n 工具設計成為可以安裝的模組	34
第四章	測試案例分析	36
4.1	新增與修改之代理關鍵字的單元測試	36
4.1.1	Alert Should Be Present	37
4.1.2	Count Values In List	39
4.1.3	Dictionaries Should Be Equal	41
4.1.4	Select From List By Label	43
4.1.5	*Page Should Contain Element	46
4.1.6	Table Should Contain	49

4.2	改善翻譯邏輯後的驗收測試示例 . . . . .	52
4.3	一詞多譯情況下所產生的圖形化使用者介面 . . . . .	55
4.4	使用涵蓋多項代理關鍵字的測試腳本 . . . . .	59
第五章	結論與未來展望 . . . . .	63
5.1	結論 . . . . .	63
5.2	i18n 工具的使用限制與未來展望 . . . . .	64
參考文獻	. . . . .	66



# 表目錄

表 3.1	本論文新增的 20 種代理關鍵字類別 . . . . .	18
表 3.2	第一版 i18n 已提供代理的 Robot Framework 原生關鍵字 . . . . .	20
表 3.3	新版 i18n 將擴充代理的 Robot Framework 原生關鍵字 . . . . .	21





# 圖目錄

圖 2.1	英文 JSON 翻譯檔示例 . . . . .	6
圖 2.2	中文 JSON 翻譯檔示例 . . . . .	6
圖 2.3	於 HTML 內加入翻譯標記 . . . . .	6
圖 2.4	第一版 i18n 的系統類別圖 . . . . .	10
圖 2.5	第一版 i18n 的系統流程 Sequence Diagram . . . . .	12
圖 2.6	Robot Framework 系統參數設定 . . . . .	12
圖 2.7	英文的 JSON 格式翻譯檔示例 . . . . .	13
圖 2.8	翻譯路徑檔部分示例 . . . . .	14
圖 3.1	新版 i18n 系統類別圖 . . . . .	17
圖 3.2	SeleniumLibrary4.0.0 版修復之編號 #1274 issue . . . . .	22
圖 3.3	input_text_into_prompt 關鍵字已在 SeleniumLibrary 版本 4.0.0 被移除 . . . . .	23
圖 3.4	xpath_should_match_x_times 關鍵字已在 SeleniumLibrary 版本 4.0.0 被 移除 . . . . .	23
圖 3.5	FindElementsProxy 的 Flow Chart . . . . .	24
圖 3.6	以列舉法定義要被翻譯的 HTML 屬性 . . . . .	27
圖 3.7	新版 i18n XPath 翻譯邏輯的 Sequence Diagram . . . . .	28
圖 3.8	一詞多譯 UI 的介面設計 . . . . .	31
圖 3.9	翻譯紀錄的介面設計 . . . . .	32
圖 3.10	產生一詞多譯 UI 的 sequence diagram . . . . .	32

圖 3.11	安裝 RF-i18n-tool 指令 . . . . .	34
圖 3.12	以 i18n 預設 JSON 翻譯檔來執行 i18n 工具 . . . . .	35
圖 3.13	以使用者提供的 JSON 翻譯檔來執行 i18n 工具 . . . . .	35
圖 3.14	JSON 翻譯檔路徑格式 . . . . .	35
圖 4.1	Alert Should Be Present 在代理關鍵字下執行的測試腳本 . . . . .	37
圖 4.2	網頁上的 alert 視窗顯示的文字 . . . . .	37
圖 4.3	Alert Should Be Equal 測試通過 . . . . .	38
圖 4.4	Count Values In List 在代理關鍵字下執行的測試腳本 . . . . .	39
圖 4.5	Count Values In List 測試腳本第一次執行通過 . . . . .	40
圖 4.6	Count Values In List 測試腳本第二次執行通過 . . . . .	40
圖 4.7	Dictionaries Should Be Equal 在代理關鍵字下執行的測試腳本 . . . . .	41
圖 4.8	Dictionaries Should Be Equal 測試腳本第一次執行通過 . . . . .	42
圖 4.9	Dictionaries Should Be Equal 測試腳本第二次執行通過 . . . . .	42
圖 4.10	Select From List By Label 在代理關鍵字下執行的測試腳本 . . . . .	43
圖 4.11	Select From List By Label 測試腳本要驗證的網頁元件 . . . . .	44
圖 4.12	Select From List By Label 測試腳本第一次執行通過 . . . . .	44
圖 4.13	Select From List By Label 測試腳本第二次執行通過 . . . . .	45
圖 4.14	*Page Should Contain Element 在代理關鍵字下執行的測試腳本 . . . . .	46
圖 4.15	*Page Should Contain Element 測試腳本要驗證的網頁元件 [22] . . . . .	47
圖 4.16	*Page Should Contain Element 測試腳本第一次執行通過 . . . . .	47
圖 4.17	*Page Should Contain Element 測試腳本第二次執行通過 . . . . .	48
圖 4.18	Table Should Contain 在代理關鍵字下執行的測試腳本 . . . . .	49
圖 4.19	Table Should Contain 測試腳本要驗證的網頁元件 . . . . .	50

圖 4.20	Table Should Contain 測試腳本第一次執行通過 . . . . .	50
圖 4.21	Table Should Contain 測試腳本第二次執行通過 . . . . .	51
圖 4.22	含有 @placeholder 屬性的 XPath 之測試腳本 . . . . .	52
圖 4.23	中文版 Microsoft 網頁 @placeholder 屬性對應的實際文字 [22] . . . . .	53
圖 4.24	待翻譯 XPath 運行在未改善翻譯邏輯時的 i18n 工具之結果 . . . . .	53
圖 4.25	待翻譯 XPath 運行在改善翻譯邏輯後的 i18n 工具之結果 . . . . .	54
圖 4.26	Test should be equal 測試腳本 . . . . .	55
圖 4.27	遭遇一詞多譯且測試通過，開啟一詞多譯 UI . . . . .	55
圖 4.28	按下 Submit 按鈕後，將翻譯寫進設定檔，同時清除翻譯選項 . . . . .	56
圖 4.29	打開翻譯紀錄介面，上面記錄著使用者翻譯選擇 . . . . .	56
圖 4.30	清除選擇的翻譯紀錄後，再次打開翻譯紀錄介面 . . . . .	56
圖 4.31	測試腳本不通過的情況下，不跳出 UI，但會顯示 warning 資訊於報表 . . . . .	57
圖 4.32	測試腳本通過且沒有遭遇一詞多譯的情況下，不跳出 UI . . . . .	58
圖 4.33	Test multiple user behaviors on Microsoft website 測試腳本實作 . . . . .	60
圖 4.34	Test multiple user behaviors on Microsoft website 測試腳本第一次執行 結果 . . . . .	61
圖 4.35	Test multiple user behaviors on Microsoft website 測試腳本第二次執行 結果 . . . . .	62
圖 5.1	Get Match Count 測試腳本 . . . . .	64
圖 5.2	Dictionry Should Contain Item 測試腳本 . . . . .	64
圖 5.3	Dictionry Should Contain Item 原生實作 . . . . .	65
圖 5.4	第一次與第二次執行測試腳本，待翻譯詞的參數紀錄變化 . . . . .	65

# 第一章 緒論

## 1.1 研究背景與動機

根據朱峻平的論文:「支援多國語言的 Robot Framework 網頁自動化驗收測試」(i18n) [1], 透過使用者提供的 JSON 格式 [2] 多國語言網頁翻譯檔, 建立出一份單字對應翻譯的翻譯路徑檔, 並在程式中定義翻譯的邏輯以及代理關鍵字, 即可在不改動 Robot Framework 測試腳本的情況下, 完成多國語言網頁自動化驗收測試。

然而, 現在的 i18n [3] 工具仍然存在著許多待改善之處, 例如:

1. 目前 i18n 工具只支援 7 種 Robot Framework [4] 原生關鍵字, 如果未來測試腳本使用到其他未支援的 Robot Framework 原生關鍵字, 便會發生錯誤。
2. 程式目前的翻譯對象僅限於網頁上的 `text()`、`normalize-space()`、`@title` 三種 HTML 屬性。屆時, 若使用者的測試腳本是用沒有列舉出來的屬性撰寫, 例如: `@placeholder`、`@arial-label` 等等, 便會出錯。
3. 測試腳本執行期間, 若遭遇一詞多譯的情況, 目前 i18n 工具僅於報表上列出該待翻譯詞的可能翻譯有哪些, 並從中選取會使測試通過的翻譯詞當作當前翻譯; 然而, 使用者無法自己選擇期望的翻譯去跑測試腳本。此外, 假如存在多個翻譯詞都會通過此測試 (可能是因為 XPath [5] [6] 使用 `contains` 語法, 使得畫面上要驗證的資訊只要包含於翻譯詞, 測試就會通過; 又或者畫面上剛好有多個元件符合翻譯後的測試腳本), 翻譯過後腳本的測試對象, 就會偏離了使用者原先的預期。

4. 目前 i18n 工具尚未包裝成可以讓使用者直接安裝後使用的擴充工具，停留在使用者必須將 github 上的專案 clone 下來，並且在該專案上開發測試腳本的階段。如此會造成使用者的許多不便。

有鑑於此，本論文將對現有的 i18n 工具進行功能的改善與擴充。

## 1.2 研究目標

本論文旨在改善 i18n 工具上述提及的四項缺陷，希望透過以下努力可以讓此 I18n 工具在未來執行多國語言網頁自動化驗收測試 [7] [8] 時，更易於使用。

實作的目標分別列舉如下：

1. 擴充完剩下的 Robot Framework 原生關鍵字代理，使得所有常用原生關鍵字的參數部分都能正確的被翻譯。
2. 撰寫出一套新的 XPath 翻譯邏輯，支援除了 @title、normalize-space()、text() 以外，其他 HTML 屬性的翻譯檢查。
3. 執行完含有一詞多譯的測試腳本後，如果測試通過，便開啟一個圖形化介面，記錄了執行翻譯時當下關鍵字的參數組合，並顯示所有可能的翻譯詞，讓使用者可以從中去選擇，並產生一個設定檔。之後再次執行測試腳本時，i18n 工具便會根據設定檔的內容去選擇適當的翻譯詞，同時消除報表上的 warning 提示。
4. 將 i18n 工具包裝成為可以 pip [9] install 的 library。

## 1.3 論文組織架構

本論文分為五章，第一章介紹研究背景與動機，以及期望達成的研究目標。第二章介紹本論文相關的背景知識。第三章探討本論文的研究方法與實作，詳述擴充代理關鍵字流程、改善翻譯邏輯的完整思路，以及如何藉由圖形化使用者介面去改善 i18n 工具目前遇到一詞多譯時的處理方法。第四章展示實際的測試案例，呈現做了上述改動後，使用者執行多國語言網頁自動化驗收測試時，會得到什麼結果，並可自行與第一版 i18n 工具 [1] 做比較。第五章則是結論以及未來展望，概述本論文的成果以及尚且存在的使用限制和待改善之處。



## 第二章 背景知識

### 2.1 i18n 工具相關知識

#### 2.1.1 國際化

國際化 (Internationalization) [3]，簡寫為 i18n，其中 18 代表 i 和 n 之間的 18 個英文字母。國際化是開發軟體時，將軟體本身和特定語言、地區脫鉤的一個過程，除了可以滿足不同的地區、文化的大眾需求，移植軟體到不同的語言環境時，也不須改變內部程式的實作。



#### 2.1.2 自動化驗收測試

驗收測試 (Acceptance Testing) [7]，是一種站在使用者的立場，去檢驗一個真實存在的系統，是否滿足使用者需求與預期的測試方法。

而透過撰寫自動化測試腳本 [10]，如今我們可以執行更符合成本，且更加精確的自動化驗收測試。改善了過往手動執行驗收測試時，存在的人為操作誤差、系統問題無法即時呈現、成本較高等問題。

### 2.1.3 XPath

XPath [5]，全名為 XML Path Language，可以用來定位 XML 檔案中某節點所處在的位置。在使用 Robot Framework 撰寫的網頁自動化驗收測試中，我們便時常需要藉助 XPath，去找到並確定畫面上某元件的位置，以對其狀態進行測試。

### 2.1.4 JSON 格式翻譯檔在前端框架中的運作

JSON 格式 [2] 的多國語言翻譯檔，其結構是由撰寫者自定義的 ‘Key’，對應網頁上所要呈現的翻譯詞 ‘Value’ 所構成；而一個 Value，可以使用一個或以上的 Key 階層結構定位。例如：在英文 JSON 翻譯檔下，i18n 這個 Value，其 Key 值是 “EXAMPLE.INTERNATIONALIZATION” (如圖 2.1)。

而在前端框架中，若我們要透過 JSON 格式的多國語言翻譯檔，進行多國語言網頁翻譯，則必須使用第三方套件；在此以 AngularJS [11] 搭配第三方套件 angular-translate [12] 為例，將英文網頁翻譯成中文網頁。首先，將待翻譯詞定義在 JSON 格式的英文翻譯檔 (如圖 2.1)，以及其中文翻譯定義在 JSON 格式的中文翻譯檔 (如圖 2.2)，並在 HTML 內的需翻譯詞處加上對應的標記 (如圖 2.3)。之後，AngularJS 便會根據當前網頁的語言-中文，以及待翻譯元件標記處的 Key 值-“EXAMPLE.INTERNATIONALIZATION”，去中文的 JSON 翻譯檔查找該 Key 值對應的 Value，最後將找到的翻譯詞-「國際化」顯示於網頁上。



```

1  {
2    | "EXAMPLE": {
3    |   | "INTERNATIONALIZATION": "i18n"
4    |   }
5    }

```

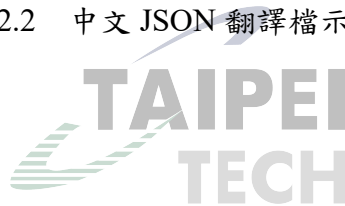
圖 2.1 英文 JSON 翻譯檔示例

```

1  {
2    | "EXAMPLE": {
3    |   | "INTERNATIONALIZATION": "國際化"
4    |   }
5    }

```

圖 2.2 中文 JSON 翻譯檔示例



```
<h1>{{'EXAMPLE.INTERNATIONALIZATION' | translate}}</h1>
```

圖 2.3 於 HTML 內加入翻譯標記

## 2.2 Robot Framework

Robot Framework [4] [13] 是一個開源的框架語言，可以用來執行自動化驗收測試或機器人自動化，核心框架是由 Python [14] 編寫而成，測試者可以使用 Python 或 Java 擴充其函式庫。其特色是擁有簡單的語法，以及容易理解的原生關鍵字 (Keyword)，測試者可以視需求使用並包裝成更接近自然語言的關鍵字。

### 2.2.1 Robot Framework 關鍵字

Robot Framework 透過許多的關鍵字 (Keyword) 來構成一份測試腳本，關鍵字分為兩種：Robot Framework 的原生關鍵字，以及使用者自己定義的關鍵字。Robot Framework 的原生關鍵字可以透過引入 Library 來使用，能夠達成絕大多數對網頁上元件的基本操作，例如：Click Element、Open Browser；或是一些基礎的邏輯判斷，例如：Should Be Equal；以及一些對變數的設定與印出，例如：Set Variable、Log。

而使用者自定義的關鍵字，則是根據當下測試案例 (Test Case) 的需要，將一或多個原生關鍵字包裝成為更具體、易讀、能夠描述當下測試動作的關鍵字；例如：想使用測試腳本去點擊 Microsoft 官方網頁上的「支援」按鈕，則可撰寫 “Click Support Button On Microsoft Page” 這個自定義的關鍵字，裡面包裝了原生關鍵字 “Open Browser” 以及 “Click Element”，分別執行打開瀏覽器和點擊按鈕的動作。

此外，實際使用上，隨著每個人或組織的要求程度不同，包裝原生關鍵字成為自定義的關鍵字也會有嚴謹程度上的不同。但是 Robot Framework 本身對這部分並沒有強制的規定，是很自由的。

### 2.2.2 Robot Framework 測試腳本

一個 Robot Framework 的測試腳本基本上由四個區塊構成，分別是：

1. Settings: 包含了使用到的 library 與 resource file，也可以將 Test Setup(測試腳本執行前要做的動作)、Test Teardown(測試腳本執行後要做的動作) 定義於此。
2. Test Cases: 在此測試者可以為各項想要驗證的使用者需求，撰寫核心的測試腳本
3. Keywords: 如果測試腳本使用並非 Robot Framework 的原生關鍵字，或未被定義於其他 resource file 中，測試者可以在此處撰寫出新的關鍵字去達到測試目的。
4. Variables: 測試者可以在此處定義測試案例需要用到的 Global 變數。

### 2.2.3 Robot Framework 測試報表

在測試腳本運行結束後，Robot Framework 會產生出一份測試報表，記錄了整體的執行狀況，包含測試通過或失敗、腳本執行時間，並透過可展開的階層式圖表，方便測試者去追蹤到當前出現問題的關鍵字，進而修復錯誤。

### 2.2.4 Robot Framework Listener

Robot Framework 提供了 Listener [13] 的介面，用以接收測試不同時期的資訊，並可以實作此介面，在測試腳本執行的不同時期做出額外的反應；例如實作 end\_suite() 函式，在測試結束後開啟一詞多譯 UI，供使用者選擇。

此外，Robot Framework 的 Listener 分為 Version2 和 Version3 兩種版本；以本論文來說，使用的是版本 Version2，必須在實作 Listener 處加上：

```
ROBOT_LISTENER_API_VERSION=2
```

以供系統辨認要實作的 Listener 版本。而為了讓系統在測試執行時呼叫 Listener，在執行測試腳本前必須設定 -listener 組態，來指定需要使用到哪些自定義的 Listener。

## 2.3 第一版 i18n 的系統介紹

以下將分成第一版 i18n 的系統架構、執行流程，以及執行測試腳本時會產生一詞多譯問題的緣由，來說明第一版 i18n 工具的相關背景知識。

### 2.3.1 第一版 i18n 的系統架構

第一版 i18n 的系統類別圖 (如圖 2.4)，包含了 3 個 Robot Framework 原生類別: RobotFramework、Listener、和 Report；第一版 i18n 所設計的 5 個系統類別: I18nListener、MappingRoutesGenerator、I18nMap、I18nTrigger、Proxy，以及 5 個實作自 Proxy [15] 類別的代理關鍵字類別: ShouldContainProxy、ElementTextShouldBeProxy、FindElementProxy、ListsShouldBeEqualProxy、ShouldBeEqualProxy。[1]



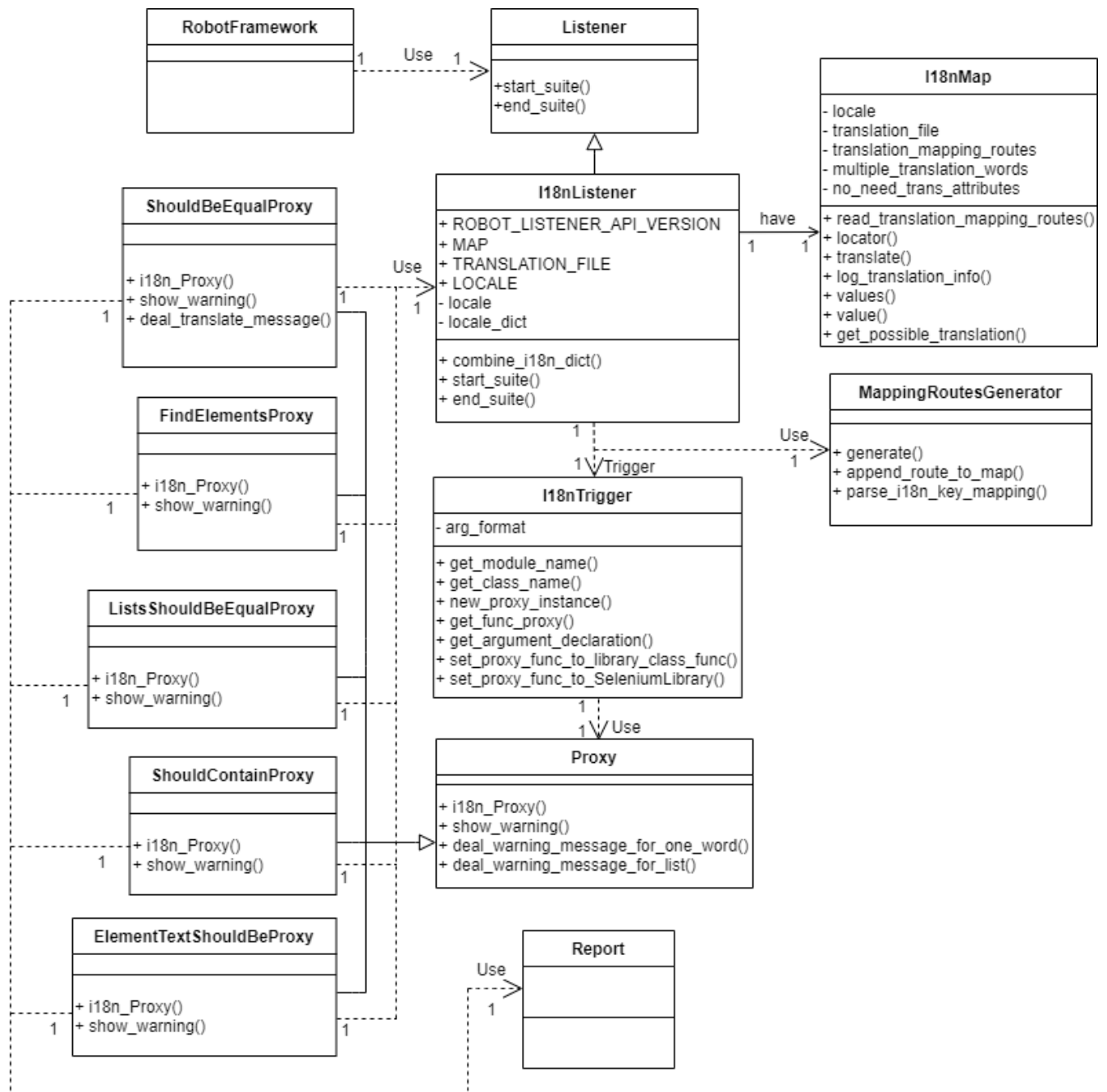


圖 2.4 第一版 i18n 的系統類別圖

I18nListener 類別實作 Listener 類別，負責程式開始與結束時執行 start\_suite() 和 end\_suite() 的內容，並對 MappingRoutesGenerator、I18nMap、I18nTrigger 三個類別進行初始化的動作。I18nMap 類別負責處理各個代理關鍵字類別透過 I18nListener 類別的呼叫，並將待翻譯詞做翻譯後回傳。I18nTrigger 類別透過 new\_proxy\_instance()

函式建立各個代理關鍵字類別的實例；並使用 `set_proxy_func_to_library_class_func()`、`set_proxy_func_to_SeleniumLibrary()` 兩函式，將代理關鍵字類別的實作包裝於 Robot Framework 原生關鍵字之外，使每次呼叫關鍵字時，必先執行代理關鍵字的實作。MappingRoutesGenerator 類別負責以英文 JSON 翻譯檔為基準，產生出翻譯路徑檔，讓測試腳本運行在其他語言環境下，可以根據此路徑，在所屬語言的 JSON 翻譯檔下找到正確的翻譯。Proxy 類別提供了一個介面，讓實作 Proxy 類別的代理關鍵字類別可以根據各自的需要，去擴充內部的實作；其中，`i18n_Proxy()` 函式提供各代理關鍵字類別撰寫核心的功能。

### 2.3.2 第一版 i18n 的系統執行流程

第一版 i18n 系統流程的 Sequence Diagram(如圖 2.5)，在使用 i18n 工具執行測試腳本前，使用者必須於 Red 編輯器 [16] 中的 Additional Robot Framework arguments 設定系統參數為 `-d out -L debug - -listener i18n/listeners/I18nListener.py:zh-TW`，zh-TW 代表當前語言為繁體中文-台灣(如圖 2.6)。並且，使用此 i18n 工具的前提，必須建立在「使用者對於自己所提供的 JSON 翻譯檔，有充分的了解」，如此，當遭遇一詞多譯時，使用者才會明確知道在當下情況，什麼翻譯是正確的。(此前提套用到新版的 i18n 工具上亦然)

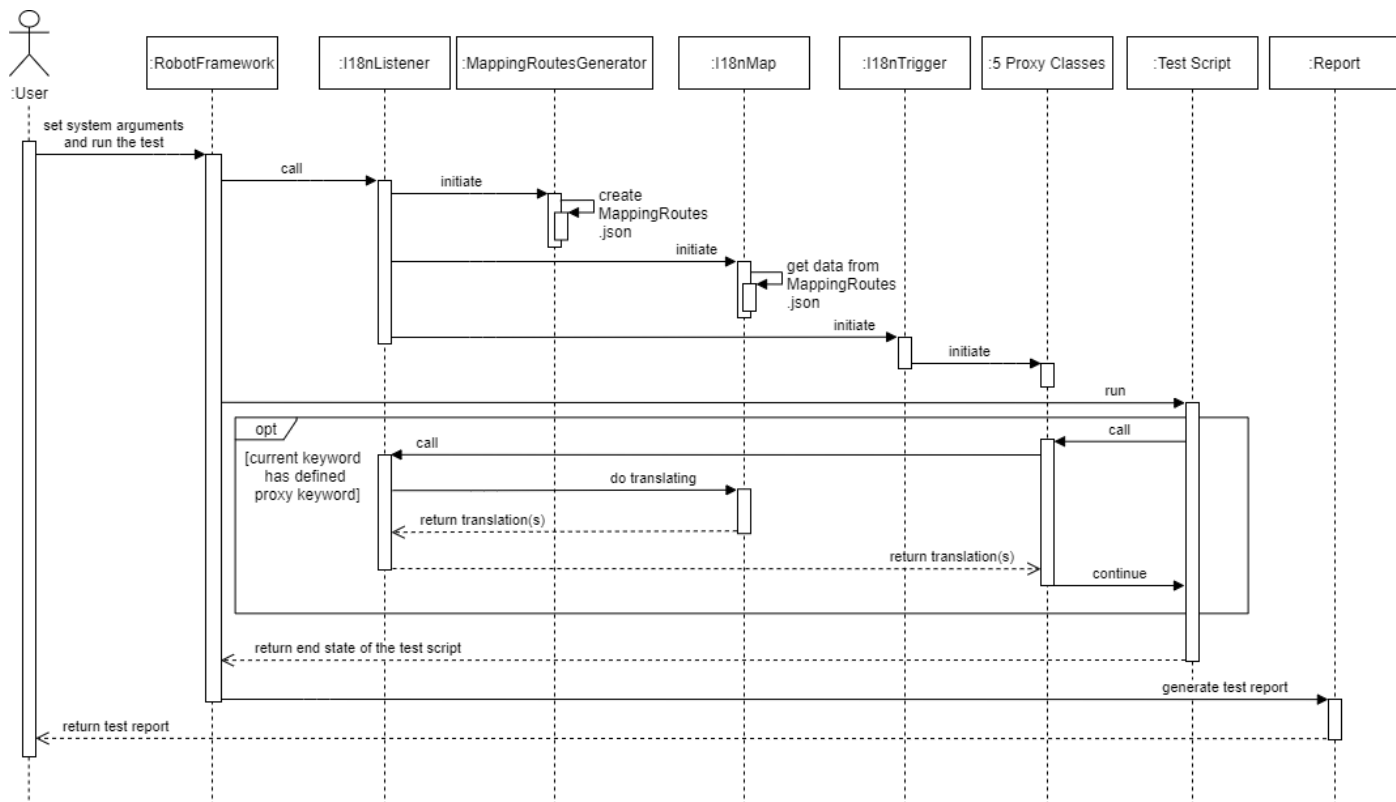


圖 2.5 第一版 i18n 的系統流程 Sequence Diagram

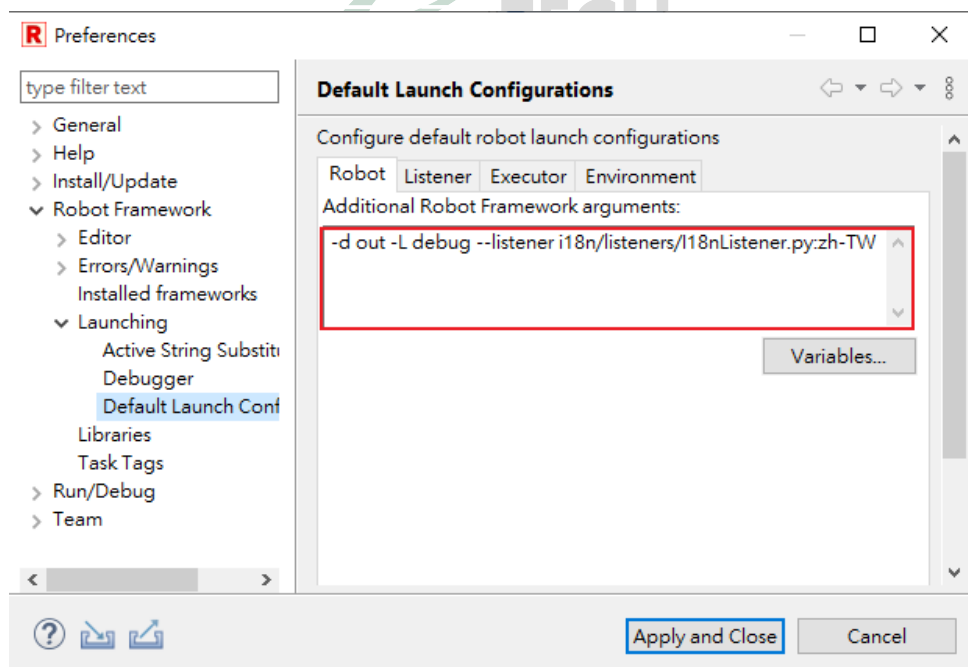


圖 2.6 Robot Framework 系統參數設定

測試執行時，系統會先去呼叫 I18nListener 類別並執行其實作。之後初始化 MappingRoutesGenerator 類別，藉由讀取由使用者自行提供的英文 JSON 格式翻譯檔 (如圖 2.7)，建立出一份由「待翻譯詞」和「Key 階層」構成的翻譯路徑檔 (如圖 2.8)；例如待翻譯詞 'Support'，有兩種 Key 階層的路徑：“['SUPPORT']”和“['SUB\_BAR'] ['BLUE\_TITLE'] ['MICROSOFT\_SUPPORT']”，代表 'Support' 在多國語言網頁中，會在不同情境下被翻譯成兩種不同的詞彙。接著系統會依序初始化 I18nMap、I18nTrigger，以及所有代理關鍵字類別。

```
listeners > languageFiles > en-US > {} common-en-US.json > ...
You, 2 weeks ago | 2 authors (You and others)
1  {
2    "OFFICE": "Office",
3    "WINDOWS": "Windows",
4    "SURFACE": "Surface",
5    "XBOX": "Xbox",
6    "DEALS": "Deals",
7    "SUPPORT": "Support",
8    "HELLO_I_AM_AN_ALERT_BOX": "Hello! I am an alert box!",
9    "Apple" : "Apple",
10   "SOMORE": "More",
11   "HOWCANIHELPU": "How can we help you?",
12   "TITLE" : "Microsoft - Official Home Page",
13   "WelcomeBing": "Welcome to Bing's website",
14   "All Microsoft": "All Microsoft",
15   "I18nWebTitle": "I18n Web Testing"
16 }
```

圖 2.7 英文的 JSON 格式翻譯檔示例



```
listeners > {} mappingRoutes.json > ...
1  {
2    "Office": [
3      "'OFFICE'"
4    ],
5    "Windows": [
6      "'WINDOWS'"
7    ],
8    "Surface": [
9      "'SURFACE'"
10   ],
11   "Xbox": [
12     "'XBOX'"
13   ],
14   "Deals": [
15     "'DEALS'"
16   ],
17   "Support": [
18     "'SUPPORT'",
19     "'SUB_BAR'['BLUE_TITLE']['MICROSOFT_SUPPORT']"
20   ],
21   "Hello! I am an alert box!": [
22     "'HELLO_I_AM_AN_ALERT_BOX'"
23   ],
24   "Apple": [
25     "'Apple'"
26   ],
27   "More": [
28     "'SOMORE'",
29     "'MORE'"

```



圖 2.8 翻譯路徑檔部分示例

當執行完上述工作後，測試腳本才會正式開始執行；每當腳本運行到一個有定義代理的關鍵字，系統就會去呼叫對應的代理關鍵字物件，並執行翻譯邏輯。最後，當測試腳本執行結束，系統則會將一詞多譯的 warning 資訊顯示在報表上。

### 2.3.3 產生一詞多譯問題的緣由

執行多國語言網頁自動化驗收測試時，會產生「一詞多譯」問題的根本原因，源自於在前端的多國語言網頁設計中，可能必須根據當下網頁情況的不同，將同一個詞語，在不同語言的網頁上翻譯成為不同的詞。例如：在 Microsoft 英文官方網頁上的‘Support’，在中文版網頁對應的不同頁面上，被分別翻譯為「支援」與「支援服務」。

此外，前端開發人員會隨著多國語言網頁的設計，產出一包 JSON 翻譯檔，裡面定義著不同語言之間，同一個詞語應該要被如何翻譯；並將此包 JSON 翻譯檔交給測試人員，以進行多國語言的網頁自動化驗收測試。

然而，對於測試人員而言，假如要驗證畫面上的一個元件，其 text() 屬性值是‘Support’的翻譯，在遭遇一詞多譯時，JSON 檔卻只能告知‘Support’可能被翻譯成「支援」或「支援服務」；JSON 檔內的 Key 階層對於測試人員而言並沒有太大的意義，因為測試腳本的撰寫是站在模擬使用者操作的角度，並不會把前端開發者的設計思維考慮進去。如此，就會造成多國語言網頁的一詞多譯特性，在測試人員進行自動化驗收測試的階段時，發生系統不知該如何選擇當下正確翻譯詞的問題。

第一版的 il8n 工具，在遭遇一詞多譯時，僅於測試報表上顯示 warning 資訊，提供待翻譯詞可能的翻譯有哪些；雖然成功呈現了系統目前有遭遇一詞多譯的問題，但尚未給出一個有效的解法。

## 第三章 新版 i18n 的系統設計

本章將詳述如何針對 i18n 工具現存的四項問題，進行思考與改善，並點出與朱峻平的論文「支援多國語言的 Robot Framework 網頁自動化驗收測試」(第一版 i18n 工具)[1] 的不同之處。

### 3.1 系統擴充

新版 i18n 系統類別圖 (如圖 3.1)，沿用第一版 i18n 工具的架構 (如圖 2.4)，經過部分實作的改善，並且新增了一個用於顯示一詞多譯選項的圖形化使用者介面類別 (UI)，以及 20 種代理關鍵字的類別。(新增的類別在圖中以紅色方框標示)

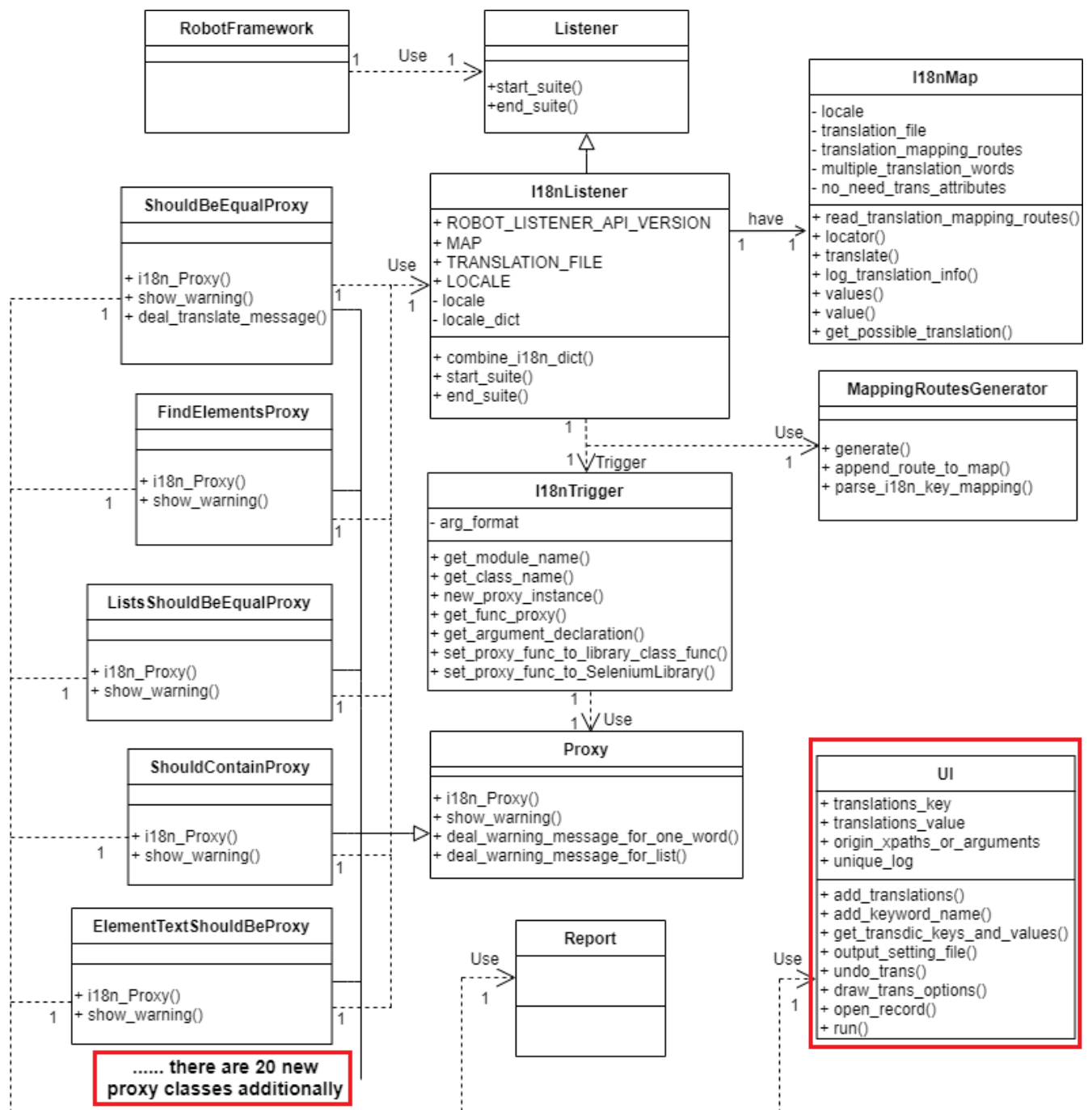


圖 3.1 新版 i18n 系統類別圖

本論文新增的 UI 類別會在程式執行期間，記錄下遭遇到一詞多譯的詞語以及其翻譯。在程式執行後，產生一詞多譯的 UI；使用者選擇並提交希望的翻譯後，便會產生一個系統設定檔記錄翻譯選擇。

而其他 20 種新增的代理關鍵字類別 (如表 3.1)，皆實作父類別 Proxy，負責代理各自對應之原生關鍵字的參數部分翻譯，並將翻譯好的參數部分回傳給對應的原生關鍵字。

表 3.1 本論文新增的 20 種代理關鍵字類別

新增的代理關鍵字類別	
1. AlertShouldBePresentProxy	11. ListShouldNotContainDuplicatesProxy
2. CountValuesInListProxy	12. RemoveFromDictionaryProxy
3. DictionariesShouldBeEqualProxy	13. RemoveValuesFromListProxy
4. DictionaryShouldContainItemProxy	14. SelectFromListByLabelProxy
5. DictionaryShouldContainKeyProxy	15. SelectFromListByValueProxy
6. DictionaryShouldContainValueProxy	16. TableCellShouldContainProxy
7. GetMatchCountProxy	17. TableColumnShouldContainProxy
8. ListSelectionShouldBeProxy	18. TableRowShouldContainProxy
9. ListShouldContainSubListProxy	19. TableShouldContainProxy
10. ListShouldContainValueProxy	20. TitleShouldBeProxy

系統執行流程相較於第一版 i18n 改動較大的部份，是在測試腳本結束後；除了將一詞多譯的 warning 資訊顯示在報表上外，若是遭遇過一詞多譯，便會跳出一詞多譯的 UI(詳見 3-4 節)，記錄了執行翻譯當下關鍵字的參數組合，並顯示所有可能的翻譯詞，讓使用者可以從中去選擇並產生一個設定檔。之後再次執行測試腳本時，系統便會根據設定檔的內容去選擇適當的翻譯詞，同時消除報表上的 warning 提示資訊。

## 3.2 擴充與修改代理關鍵字實作

以下將分為新版 i18n 需要擴充的代理關鍵字、未來遭遇 Robot Framework 相關 Library 版本更新的應對，以及新版 i18n 擴充代理關鍵字的實作步驟來做說明。

### 3.2.1 新版 i18n 需要擴充的代理關鍵字

我們需要為一個 Robot Framework 的原生關鍵字撰寫代理關鍵字，有以下兩點情況：

1. 關鍵字的參數部分包含純文字部分，所以有翻譯的需求，可能是基本的 string、list、dictionary；例如：關鍵字 `should_be_equal`，其參數部分代入 'Support'，'Support'，去驗證兩者是否相同。
2. 關鍵字的參數部分包含 XPath，且 XPath 內的 HTML 屬性會包含純文字的部分，所以有翻譯的需求；例如：關鍵字 `find_element`，其參數部分代入 XPath `"//*[ @id = 'supHomeAndLandingPageHeaderContainer' ]//*[ contains(text(), 'Support' )]"`，去驗證畫面上的橫向標題 container 是否包含 'Support' 字樣。

根據朱峻平的論文 [1]，第一版 i18n 工具只支援 7 種 Robot Framework 原生關鍵字 (如表 3.2)；並整理出尚有 31 種參數部分需要翻譯的原生關鍵字，未提供支援。如果測試腳本使用到這些未支援的原生關鍵字去執行多國語言網頁測試，則會導致出錯。因此，本論文的解法是擴充完目前 Robot Framework 版本剩下的代理關鍵字，使得有翻譯需求的原生關鍵字，其參數部分都能正確的被翻譯。

表 3.2 第一版 i18n 已提供代理的 Robot Framework 原生關鍵字

第一版 i18n 已提供代理的關鍵字
1. find_element
2. element_text_should_be
3. lists_should_be_equal
4. should_be_equal
5. should_not_be_equal
6. should_contain
7. should_not_contain

且又因為先前的 i18n 工具分別在 Robot Framework 版本 3.0.4、SeleniumLibrary 版本 3.1.1 下開發，而現在的 Robot Framework 版本已更新到 3.2.2，SeleniumLibrary 版本則更新到 4.5.0，導致原先需要被支援的 31 種原生關鍵字，有兩種已被淘汰：input\_text\_into\_prompt、xpath\_should\_match\_x\_times。因此，還有 29 種原生關鍵字需要提供代理 (如表 3.3)。(‘\*’ 標示為因版本更新導致被淘汰的原生關鍵字)

表 3.3 新版 i18n 將擴充代理的 Robot Framework 原生關鍵字


待擴充的代理關鍵字	
Collections Libaray (3.2.2)	SeleniumLibrary (4.5.0)
1. count_values_in_list	1. alert_should_be_present
2. dictionaries_should_be_equal	2. input_text_into_alert
3. dictionary_should_contain_item	3. *input_text_into_prompt
4. dictionary_should_contain_sub_dictionary	4. *xpath_should_match_x_times
5. dictionary_should_contain_key	5. list_selection_should_be
6. dictionary_should_not_contain_key	6. select_from_list_by_label
7. dictionary_should_contain_value	7. unselect_from_list_by_label
8. dictionary_should_not_contain_value	8. select_from_list_by_value
9. list_should_contain_sub_list	9. unselect_from_list_by_value
10. list_should_contain_value	10. title_should_be
11. list_should_not_contain_value	11. table_should_contain
12. list_should_not_contain_duplicates	12. table_header_should_contain
13. remove_from_dictionary	13. table_footer_should_contain
14. remove_values_from_list	14. table_cell_should_contain
15. get_match_count	15. table_column_should_contain
	16. table_column_should_contain



### 3.2.2 未來遭遇 Robot Framework 相關 Library 版本更新的應對

未來若又經歷了 Robot Framework 和 SeleniumLibrary 版本的更新，可以透過查閱 Robot Framework 和 SeleniumLibrary 在官方 github 上 [17] [18] 的 release notes 來了解版本變更資訊，並對新版本新增/修改/刪除的原生關鍵字進行代理關鍵字實作上的更新。在此將以實際的操作步驟，說明本論文是如何透過查閱 SeleniumLibrary 官方 github 上的 release notes，發現 `input_text_into_prompt` 和 `xpath_should_match_x_times` 關鍵字已被淘汰。

首先，到 SeleniumLibrary 的 release 頁面，因為本次 i18n 工具是從 SeleniumLibrary 的版本 3.1.1 升至 4.5.0，所以從 3.1.1 版一個個往上找，最後於版本 4.0.0 的 release notes 處發現了 issue 編號 #1274 “Remove keywords which where officially deprecated in previos releases” 有異樣 (如圖 3.2)，點擊連結進入此 issue；進一步發現 issue 編號 #1334，再次點擊進入，並在 Files changed 頁面中，發現 `input_text_into_prompt` 和 `xpath_should_match_x_times` 關鍵字已在此次更新後被移除 (如圖 3.3、圖 3.4)。

A screenshot of a GitHub issue page for SeleniumLibrary. The issue is titled "Remove keywords which where officially deprecated in previos releases." and is categorized as an "enhancement" with a "medium" priority. The issue number "#1274" is highlighted in blue. The word "Remove" is highlighted in yellow.

#1274	enhancement	medium	Remove keywords which where officially deprecated in previos releases.
-------	-------------	--------	--

圖 3.2 SeleniumLibrary4.0.0 版修復之編號 #1274 issue

```

31 - @keyword
32 - def input_text_into_prompt(self, text):
33 -     """*DEPRECATED in SeleniumLibrary 3.2.* Use `Input Text Into
    Alert` instead.
34 -
35 -     Types the given ``text`` into an input field in an alert.
36 -     Leaves the alert open.
37 -     """
38 -
39 -     self.input_text_into_alert(text, self.LEAVE)

```

圖 3.3 input\_text\_into\_prompt 關鍵字已在 SeleniumLibrary 版本 4.0.0 被移除

```

997 - @keyword
998 - def xpath_should_match_x_times(self, xpath, x, message=None,
    loglevel='TRACE'):
999 -     """*DEPRECATED in SeleniumLibrary 3.2.* Use `Page Should
    Contain Element` with ``limit`` argument instead."""
1000 -     self.locator_should_match_x_times('xpath:'+xpath, x, message,
    loglevel)

```

圖 3.4 xpath\_should\_match\_x\_times 關鍵字已在 SeleniumLibrary 版本 4.0.0 被移除

### 3.2.3 新版 i18n 擴充代理關鍵字的實作步驟

以下將以 FindElementsProxy 的 Flow Chart 為例，詳述現今配合新增的一詞多譯 UI 下，本論文是如何完成代理關鍵字的擴充和修改。(沿用第一版 i18n 的部分會以 ‘\*’ 號標註)

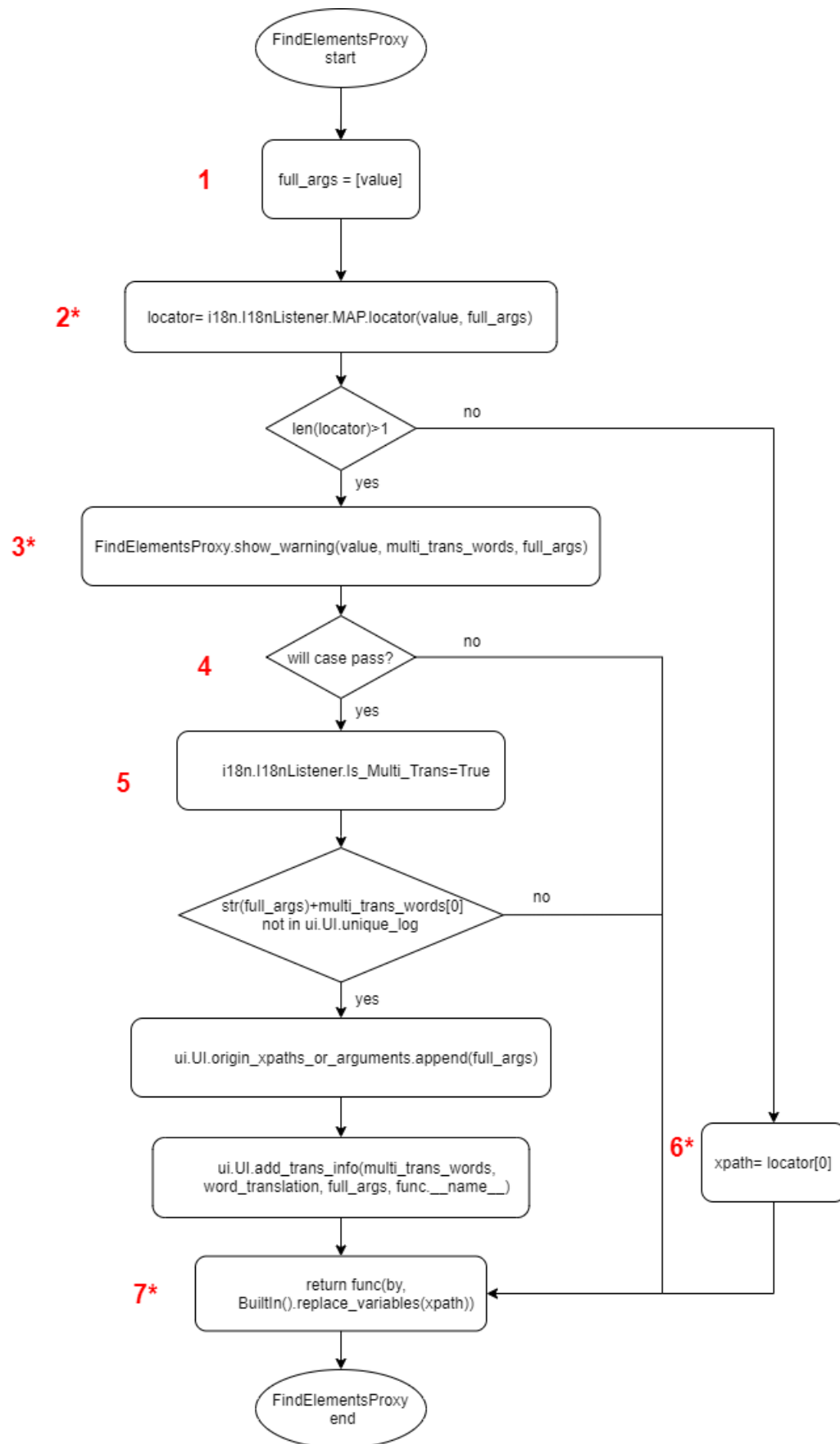


圖 3.5 FindElementsProxy 的 Flow Chart

由圖 3.5 FindElementProxy 的 Flow Chart 可以得知，在新的 i18n 版本下，擴充一個新的代理關鍵字必須遵從著以下步驟:(以下編號可對應到 Flow Chart 的數字)

1. 創出該次關鍵字呼叫的參數紀錄變數，full\_args。此變數之後會成為判斷當下待翻譯詞是否「已被使用者選擇翻譯」的依據。並於測試腳本結束後，顯示於一詞多譯 UI 上，最後隨著使用者的選擇一併存入設定檔內。

- 2\*. 執行翻譯。如:

```
locator = i18n.I18nListener.MAP.locator(value,full_args)
```

透過 i18nListener 類別的 MAP 變數 (其內裝著 i18nMap 類別的初始化資訊)，去呼叫 i18nMap 類別的 locator 函式，對待翻譯詞 value 執行翻譯，且必須代入 full\_args 以實現 1. 的內容。

- 3\*. 若判定此代理關鍵字遭遇一詞多譯，則產生 warning 資訊於報表上，如:

```
FindElementsProxy.show_warning(value,multi_trans_words,full_args)
```

4. 之後判斷此一詞多譯情況，最後會使測試通過或是失敗，如:

```
is_actual = BuiltIn().run_keyword_and_return_status('Get WebElement', translation_locator)
if is_actual:
```

5. 若測試會通過，則對預計開啟的 UI 做一些資料的準備，如:

```
i18n.I18nListener.Is_Multi_Trans=True
if str(full_args)+multiple_translation_words[0] not in ui.UI.unique_log
ui.UI.origin_xpaths_or_arguments.append(full_args)
ui.UI.add_trans_info(self, multiple_translation_words, word_translation, full_args, func.__name__)
```

先設定再測試結束後，需要開啟一詞多譯 UI。若該「參數 + 翻譯詞」組合尚未被翻譯過，則記錄下該次翻譯的參數部分和翻譯資訊。

6\*. 若此次代理關鍵字沒有遭遇一詞多譯，則如：

```
xpath = locator[0]
```

將第一筆翻譯同時也是唯一一筆翻譯指派給之後要回傳的 xpath。

7\*. 將翻譯好的參數部分回傳給 Robot Framework 原生關鍵字，如：

```
return func(self, by, BuiltIn().replace_variables(xpath))
```

其他未能一一介紹的代理關鍵字，儘管彼此間實作仍然存在著差異，但都是遵循著以上的架構去設計。



### 3.3 改善 XPath 翻譯邏輯使其能應對各種 HTML 屬性

在第一版 i18n 的翻譯邏輯中，若一個 XPath 內部存在多種 HTML 屬性，i18n 工具使用了列舉法 (如圖 3.6)，僅提供了 @title、text()、normalize-space() 三種屬性的翻譯規則。屆時，若測試腳本的 XPath 是用沒有列舉出來的屬性撰寫，但卻有翻譯的需求，例如:@placeholder、@arial-label 等等，則會導致測試出錯。

```
default_rule = {
  '((text|normalize-space)\(((text\(\))?\)\) ?= ?(\'|\"))([0-9a-zA-Z.?&()| | ]+)(\'|\"))': 4,
  '((text|normalize-space)\(((text\(\))?\)\) \, ?(\'|\"))([0-9a-zA-Z.?&()| | ]+)(\'|\"))': 4,
  '((@title) ?= ?(\'|\"))([0-9a-zA-Z.?&()| | ]+)(\'|\"))' : 3,
  '((@title), ?(\'|\"))([0-9a-zA-Z.?&()| | ]+)(\'|\"))' : 3
}
```

圖 3.6 以列舉法定義要被翻譯的 HTML 屬性

為了解決以上問題，本論文最後決定採用「負面表列法」，將確定不會執行翻譯的 HTML 屬性在程式中用 list 的方式儲存。若 XPath 中的 HTML 屬性不在該 list 中，則都去執行翻譯檢查。

此種作法相較於把所有 HTML 屬性都執行翻譯檢查，顧及到了系統執行效能，來的更加省時。相較於設計一個複雜邏輯，來訓練系統自行判斷當前 HTML 屬性是否要翻譯，在實作上則更為簡單，且把翻譯的決定權交給了測試者，而非全部靠系統。相較於第一版 i18n，把翻譯規則用列舉法固定，此作法則更能顧及到未來測試腳本的不確定性，因為我們無法預期使用者會用怎麼樣的方式去編寫 XPath。

以下將以新版 i18n XPath 翻譯邏輯的 Sequence Diagram 為例 (如圖 3.7)，輔以部分程式碼，詳述系統是如何根據當下情況，產生一套新的翻譯邏輯，以解決先前某些 HTML 屬性無法被翻譯的問題。

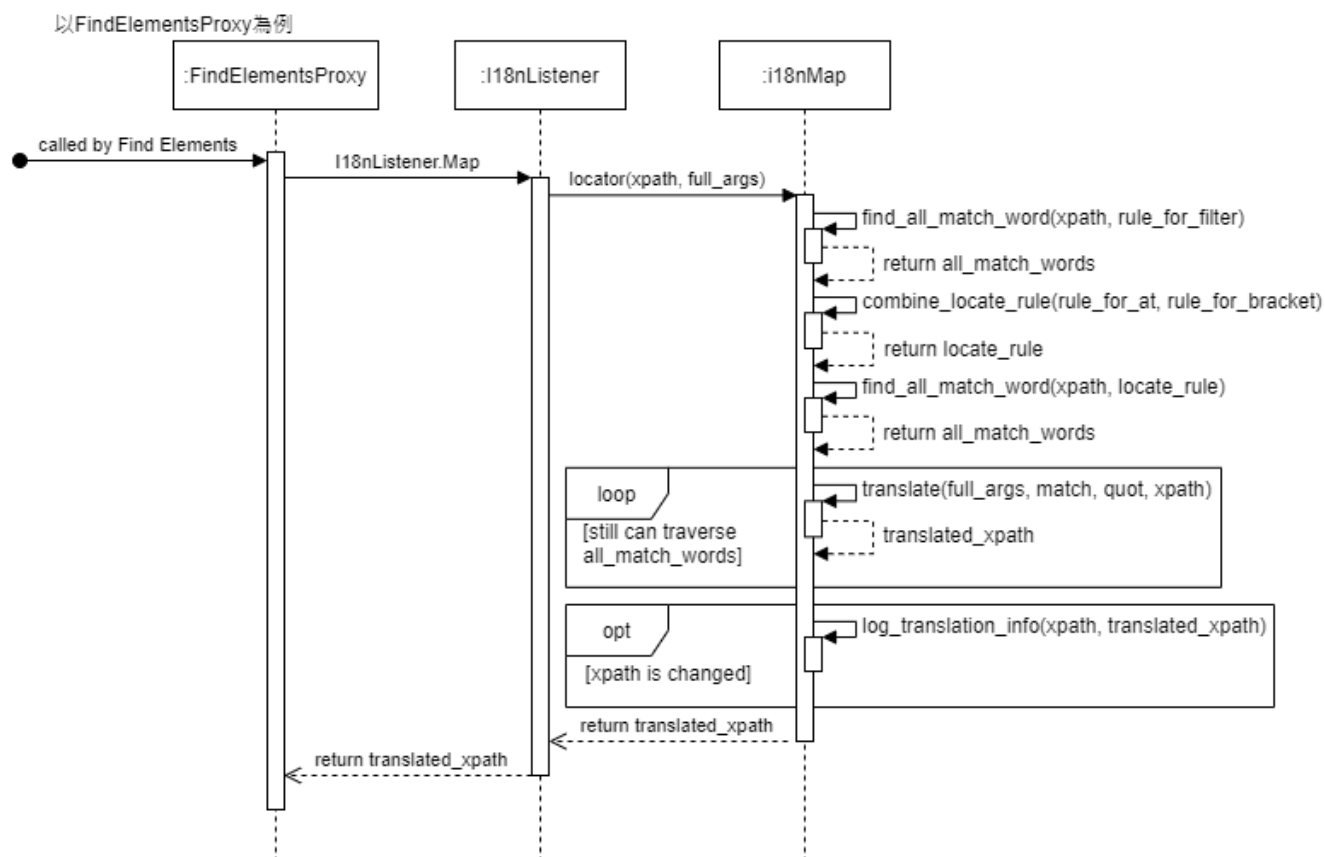


圖 3.7 新版 i18n XPath 翻譯邏輯的 Sequence Diagram

首先，被呼叫後，FindElementsProxy 透過 I18nListener 類別，呼叫 I18nMap 類別的 locator() 函式執行 XPath 的翻譯。之後，I18nMap 先定義一個內含 HTML 屬性特徵的 rule\_for\_filter 變數，如：

```
rule_for_filter = {
    "(@[a-z-]*)": "@",
    "([a-z-]*\\(\\)": "("
}
```

隨後呼叫 find\_all\_match\_word() 函式，來過濾出 XPath 中包含的所有屬性。

接著，先判斷這些屬性是否包含在「不需被翻譯屬性」的 list 中。如：

```
for rule, matches in all_match_attributes.items():
    for match in matches:
        if match not in self.no_need_trans_attrirbutes:
```

若不包含其中，則根據屬性含有 '@' 或 '()'，分別分配給 rule\_for\_at 和 rule\_for\_bracket 變數，並呼叫 combine\_locate\_rule() 函式，得到一套新的翻譯邏輯，locate\_rule。

之後，再次呼叫 find\_all\_match\_word() 函式，得到所有符合新翻譯邏輯的 HTML 屬性，all\_match\_words。並利用迴圈遍歷 all\_match\_words 中的屬性，執行 translate() 函式，進行翻譯檢查。

接著，判斷經過翻譯後，原先的 xpath 是否發生變化；例如 xpath 從原本的 `//*[text()='Software']`，變成了翻譯後的 `//*[text()='軟體']`。若 xpath 改變，則執行 log\_translation\_info() 函式，將翻譯資訊顯示於測試報表上。

最後，將翻譯後的 translated\_xpath，回傳給 FindElementsProxy 類別，完成一次 XPath 的翻譯。





### 3.4 提供圖形化使用者介面解決一詞多譯的問題

先前遇到一詞多譯問題時，第一版 il8n 工具提供了 warning 資訊於測試結束後的報表上，此作法確實提醒了使用者該腳本存在著一詞多譯問題，且也顯示出目前系統採用的翻譯詞，這是好的部分，本論文將繼續沿用。此翻譯邏輯是站在一個「最大化讓測試腳本通過」的立場，當遭遇一詞多譯時，系統會將所有的翻譯都執行看看，直到一包可能的翻譯中，出現了一個會使測試通過的翻譯，則算測試通過，並將該翻譯詞呈現於報表上。

然而，以上作法卻存在著一個缺陷，即是「使用者無法自由的選擇真正想要的翻譯詞」去執行腳本測試，而把決定全權授予系統。並有機會產生以下兩點問題：

1. 假如存在「多個翻譯都會使測試通過」，翻譯過後的腳本測試對象，就有機會偏離使用者原先的預期。其原因可能是 XPath 使用 contains 撰寫，使得翻譯只要包含特定詞即會讓測試通過。如：翻譯過後的 XPath 為 `//*[ @id= 'supHomeAndLandingPageHeaderContainer']//*[contains(@text, '支援')]`，而畫面上待驗證的元件顯示的詞是「支援服務」，測試會通過，但遺憾的是使用者預期 'Support' 在此處應該要被翻譯為「支援服務」。

又或是畫面上同時存在含有不同翻譯的元件，且滿足測試腳本的 XPath，而導致測試通過。如：畫面上兩個元件，分別顯示「支援」與「支援服務」；原本使用者預期要驗證的是畫面上存在「支援服務」字樣，但因為 'Support' 先被系統檢查的翻譯為「支援」，導致翻譯後的 XPath `//*[ @text= '支援' ]` 率先通過了測試，而不會再去檢驗後續的翻譯。

2. 假如此測試腳本原先就會發生錯誤，經過翻譯後的 XPath 卻因為碰巧滿足畫面上的某個元件，而導致測試通過。在此特殊情形下，翻譯過後的測試對象，則

更明顯的偏離了使用者原先的預期，且也改變了測試結果。如：原腳本要驗證 `//*[@text='Service']`，測試會失敗，但經過翻譯後 XPath 變成 `//*[@text= '支援服務']`，測試卻會通過，因為畫面上剛好存在「支援服務」字樣。

所以，本論文將提供一個圖形化使用者介面，記錄下執行翻譯時當下的關鍵字參數組合，並顯示所有可能的翻譯詞，讓使用者從中去選擇，並產生一個設定檔，以便之後再次執行同一測試腳本時，根據設定檔的內容去選擇適當的翻譯詞。期望透過如此的方法來改善上述兩點一詞多譯所遭遇的問題。

以下，將透過圖表和 Sequence Diagram 的方式，來詳述一詞多譯 UI 的介面設計與整體實作：

一詞多譯 UI 的介面設計 (如圖 3.8)，本論文將其設計成兩個區塊，上半部以條列的方式呈現完整的翻譯資訊，包含關鍵字名稱、參數部分、待翻譯詞、可能的翻譯。其中，參數部分記錄了當前關鍵字接受的使用者傳入參數，以作為和其他相同待翻譯詞的區別；因為相同的待翻譯詞，在不同情況下，可能擁有不同的翻譯。下半部包含一行功能資訊的提示、一個用於開啟翻譯紀錄頁面的 TransRecord 按鈕，以及提交使用者選擇並寫入設定檔的 Submit 按鈕。

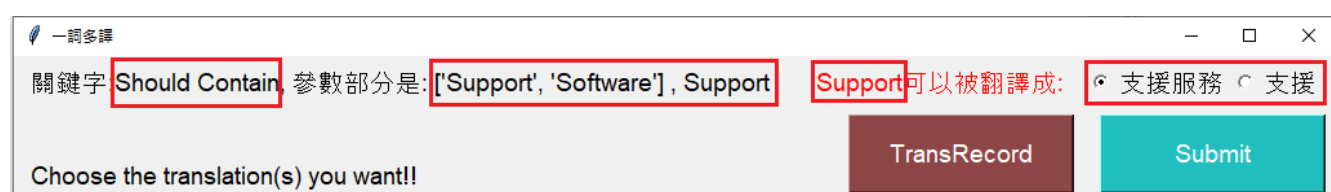


圖 3.8 一詞多譯 UI 的介面設計

翻譯紀錄的介面設計 (如圖 3.9)，本論文同樣將其設計成兩個區塊，上半部以條列式呈現設定檔中現存的使用者翻譯選擇。下半部包含一個 Undo 按鈕，考量到使用者可能不小心選錯了翻譯，用於將使用者的選擇從設定檔中刪除。



圖 3.9 翻譯紀錄的介面設計

一詞多譯 UI 的實作，大致可以分成 8 個部分: `run()`、`draw_trans_options()`、`get_transdic_keys_and_values()`、`open_record()`、`undo_trans()`、`output_setting_file()`、`add_trans_info()`、`add_keyword_name()`。各實作彼此之間的互動關係，則如圖 3.10:

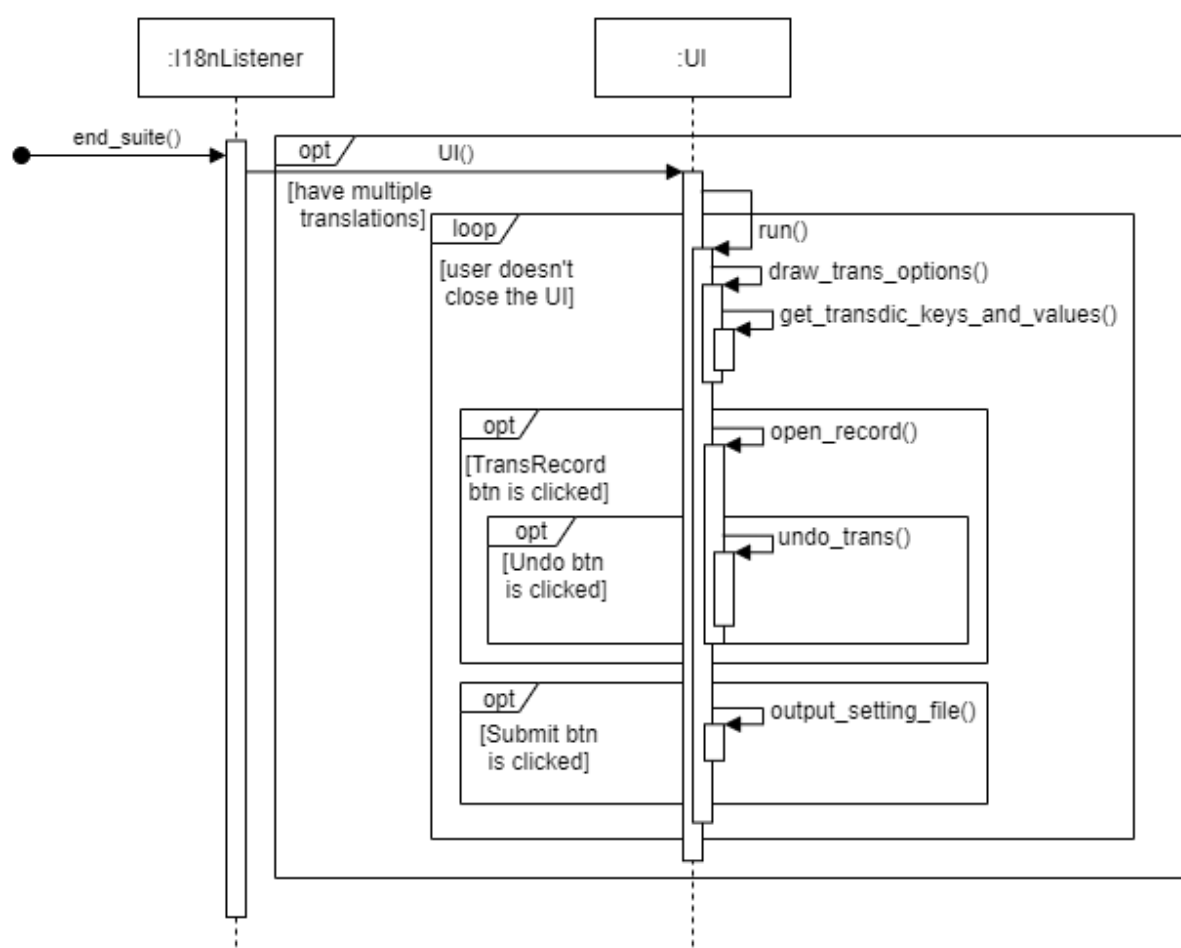


圖 3.10 產生一詞多譯 UI 的 sequence diagram

當測試腳本結束時，會呼叫 I18nListener 類別的 end\_suite() 函式。

若本次測試腳本遭遇一詞多譯，則呼叫 UI 類別，使其初始化並執行 run() 函式，產生一詞多譯 UI 的介面，包含 “Choose the translation(s) you want!!” 字樣、顯示翻譯紀錄的 TransRecord 按鈕，以及寫入使用者選擇的 Submit 按鈕。之後，run() 函式內部會呼叫 draw\_trans\_options() 函式，其內會再呼叫 get\_transdic\_keys\_and\_values() 函式，取出待翻譯詞與其對應的翻譯，最後隨著翻譯當下的關鍵字名稱和參數部分一併顯示於一詞多譯 UI 的介面上。

在一詞多譯介面上，點擊 TransRecord 按鈕會呼叫 open\_record() 函式，打開翻譯紀錄介面，其會將先前使用者選擇過的翻譯詞，以條列式的方式呈現，且介面上還包含一顆 Undo 按鈕。點擊 Undo 按鈕會呼叫 undo\_trans() 函式，將使用者選擇的翻譯紀錄從設定檔中清除掉，同時關閉翻譯紀錄頁面。

在一詞多譯介面上，點擊 Submit 按鈕則會呼叫 output\_setting\_file() 函式，將使用者選擇的待翻譯詞對應其翻譯以及完整參數寫入設定檔 “i18n/listeners/setting.txt” 中，同時將剛剛已寫入的翻譯資訊從一詞多譯 UI 上隱藏。

此外，在 UI 類別中定義的 add\_translations() 和 add\_keyword\_name() 函式，雖然不會在一詞多譯 UI 執行期間被呼叫，但在測試腳本執行期間，卻能負責將代理關鍵字傳入的關鍵字名稱、待翻譯詞，與其對應翻譯，分別儲存於變數中，方便之後一詞多譯 UI 的讀取。並且，還會將關鍵字的參數部分和待翻譯詞結合，成為可以辨別「該次測試腳本是否執行過相同翻譯」的字串，存入 unique\_log 變數中。

### 3.5 將 i18n 工具設計成為可以安裝的模組

本論文透過 Python 的 build 模組，將 i18n 工具包裝成名為“RF-i18n-tool”的 Library，並透過 twine 模組上傳至 PYPI [19] 網站上，使之成為可以直接讓使用者安裝的 Python 模組。解決了先前使用者只能從 github 上將 i18n 工具 clone 下來，並在該專案上開發測試腳本的不方便。

以下將詳述將 i18n 包裝成 Python 模組的步驟：

在將 i18n 工具打包成為一個 Library 前，必須先準備以下五個檔案：setup.cfg、pyproject.toml、README.md、LICENSE [20]、MANIFEST.in。此外必須確認將打包的資料夾下，包含 \_\_init\_\_.py 檔案，以被系統辨識為一個 Python 模組。

上述的檔案準備就緒後，便可以在 i18n 專案目錄下執行 `python -m build` 指令，待建置完畢，便會產生一個名為 dist 的資料夾，裡面裝著打包好的 RF-i18n-tool 模組檔案。接下來執行 `python -m twine upload -r repository pypi dist/*` 指令，輸入帳號密碼後，便能成功將 dist 資料夾中的 RF-i18n-tool 模組上傳至 PYPI 了。之後在 PYPI 網站上搜尋 RF-i18n-tool，便能看到上傳的模組。

使用者若要安裝此模組，只需執行 `pip install RF-i18n-tool` 指令即可 (如圖 3.11)。且之後會持續修復 bug，並推出新的版本，因此版本數字僅供參考。

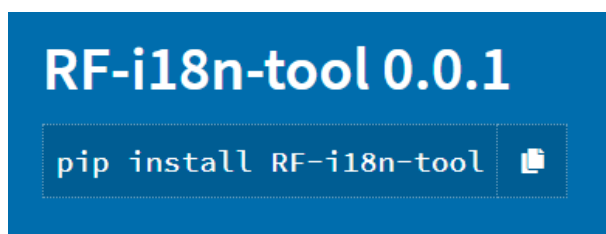


圖 3.11 安裝 RF-i18n-tool 指令

最後，使用者便可透過在 Additional Robot Framework arguments 中設定系統參數 (如圖 3.12、3.13)，來使用 i18n 工具。自定義的 JSON 翻譯檔路徑必須遵守以下格式 (如圖 3.14)

```
-d out -L debug --listener %YOUR_PYTHON_PATH%/Lib/site-packages/i18n/  
listeners/I18nListener.py:YOUR_LOCALE:i18njson
```

圖 3.12 以 i18n 預設 JSON 翻譯檔來執行 i18n 工具

```
-d out -L debug --listener %YOUR_PYTHON_PATH%/Lib/site-packages/i18n/  
listeners/I18nListener.py:YOUR_LOCALE
```

圖 3.13 以使用者提供的 JSON 翻譯檔來執行 i18n 工具

```
YOUR_PROJECT_DIR/languageFiles/YOUR_LOCALE(ex:zh-TW)/xxxYOUR_LOCALE.json
```

圖 3.14 JSON 翻譯檔路徑格式

## 第四章 測試案例分析

本章將為擴充的代理關鍵字撰寫單元測試 [21]，並且為改善翻譯邏輯與一詞多譯功能後的 i18n 工具，做功能的展示。最後將 i18n 工具套用至一個使用到多項代理關鍵字的 Robot Framework 測試腳本，以呈現「多國語言網頁自動化驗收測試」的效果。

### 4.1 新增與修改之代理關鍵字的單元測試

以下將以 Microsoft 中文官方網頁 [22]，以及筆者使用 Node.js [23] 自行架設的 i18n 測試網頁 (i18n Testing Website) 為受測網站，為各個新增或已修改實作之代理關鍵字，分別提供 Robot Framework 的單元測試腳本，以驗證原生關鍵字在套用新的代理關鍵字下，能夠獨立正常運作。但為求精簡，本論文將不重複放上同類型關鍵字的測試腳本。(經修改的第一版 i18n 代理關鍵字前將加上 ‘\*’ 標示)

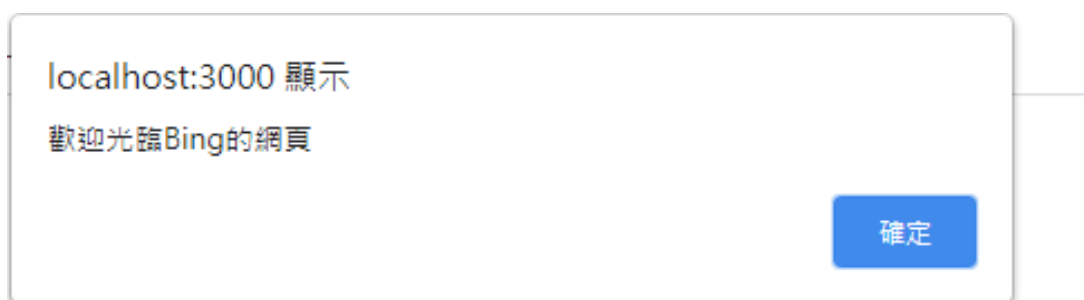
### 4.1.1 Alert Should Be Present

Alert Should Be Present 的單元測試腳本 (如圖 4.1) 會打開 i18n Testing Website，並點擊 Show Alert Message 按鈕，使畫面跳出 alert 視窗 (如圖 4.2)。最後使用 Alert Should Be Present 來驗證 alert 資訊是否為“歡迎光臨 Bing 的網頁”。測試通過，代表參數“Welcome to Bing’s website”有成功被翻譯為“歡迎光臨 Bing 的網頁” (如圖 4.3)。

```
*** Settings ***
Library      SeleniumLibrary
Library      ../self_util.py
Test Setup    Run Keywords      Open Browser      http://localhost:3000
              Chrome
...           AND      Maximize Browser Window
Test Teardown Close Browser

*** Test Cases ***
Alert should be present
    Wait until Element Is Visible    /*[normalize-space()='Show Alert
        Message']    timeout=${shortPeriodOfTime}
    Double Click Element    /*[normalize-space()='Show Alert Message']
    Alert Should Be Present    Welcome to Bing's website
```

圖 4.1 Alert Should Be Present 在代理關鍵字下執行的測試腳本



displaying the alert message, double click the "Show Alert Message" button:

Show Alert Message

圖 4.2 網頁上的 alert 視窗顯示的文字



[-] <b>TEST</b>	<b>Alert should be present</b>	00:00:03.814
<b>Full Name:</b> I18N.I18N.RegressionTest.Alert Should Be Present.Alert should be present		
<b>Start / End / Elapsed:</b> 20210605 17:20:50.023 / 20210605 17:20:53.837 / 00:00:03.814		
<b>Status:</b> <b>PASS</b> (critical)		
[+] <b>SETUP</b>	BuildIn.Run Keywords Open Browser, http:localhost:3000, Chrome, AND, Maximize Browser Window	00:00:01.398
[+] <b>KEYWORD</b>	SeleniumLibrary.Wait Until Element Is Visible //[normalize-space()='Show Alert Message'], timeout=3s	00:00:00.025
[+] <b>KEYWORD</b>	SeleniumLibrary.Double Click Element //[normalize-space()='Show Alert Message']	00:00:00.276
[+] <b>KEYWORD</b>	SeleniumLibrary.Alert Should Be Present Welcome to Bing's website	00:00:00.005
[+] <b>TEARDOWN</b>	SeleniumLibrary.Close Browser	00:00:02.108

圖 4.3 Alert Should Be Equal 測試通過



### 4.1.2 Count Values In List

Count Values In List 的單元測試腳本 (如圖 4.4)，一開始會利用 Create List 創出一個包含「支援」和「Software」的 list，並使用 Count Values In List 來計算參數「Support」出現在此 list 中幾次。第一次執行測試通過 (如圖 4.5)，透過 Log 印出的數字為 1，代表參數「Support」被翻譯為「支援」後，出現在 list 中一次，所以測試通過；且系統將支援當成「Support」翻譯的其中一種，並跳出一詞多譯 UI。之後，使用者選擇「支援」作為唯一翻譯。第二次測試通過 (如圖 4.6)，透過 Log 印出的數字為 1，且不跳出一詞多譯 UI 與 warning 資訊。

```
*** Test Cases ***
Count values in list
    @list1 = Create List      支援      Software
    ${number} = Count Values In List    ${list1}    Support
    Log      ${number}
```

圖 4.4 Count Values In List 在代理關鍵字下執行的測試腳本



圖 4.5 Count Values In List 測試腳本第一次執行通過

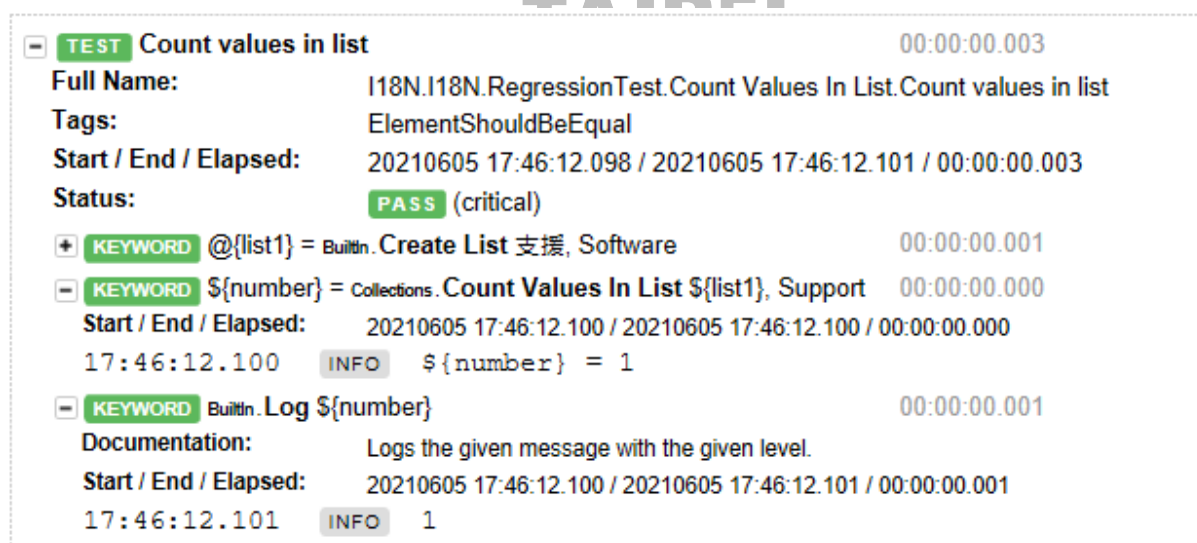


圖 4.6 Count Values In List 測試腳本第二次執行通過

### 4.1.3 Dictionaries Should Be Equal

Dictionaries Should Be Equal 的單元測試腳本 (如圖 4.7)，會利用 Create Dictionary 創出兩個 dictionaries，分別是 {'軟體': '支援'} 和 {'Software': 'Support'} (測試腳本的 '=' 左側為 key，右側為 value)，並使用 Dictionaries Should Be Equal 檢查兩者是否相同。第一次執行測試通過 (如圖 4.8)，代表 dictionary {'Software': 'Support'} 有成功被翻譯，且系統將「支援」當成“Support”翻譯的其中一種，並跳出一詞多譯 UI。之後，使用者選擇「支援」作為唯一翻譯。第二次測試通過 (如圖 4.9)，且不跳出一詞多譯 UI 與 warning 資訊，代表'Support'根據使用者選擇成功被翻譯為「支援」。

```
*** Test Cases ***
Dictionaries should be equal
&{dict1} =      Create Dictionary      軟體=支援
&{dict2} =      Create Dictionary      Software=Support
Dictionaries Should Be Equal    ${dict1}    ${dict2}
```



圖 4.7 Dictionaries Should Be Equal 在代理關鍵字下執行的測試腳本

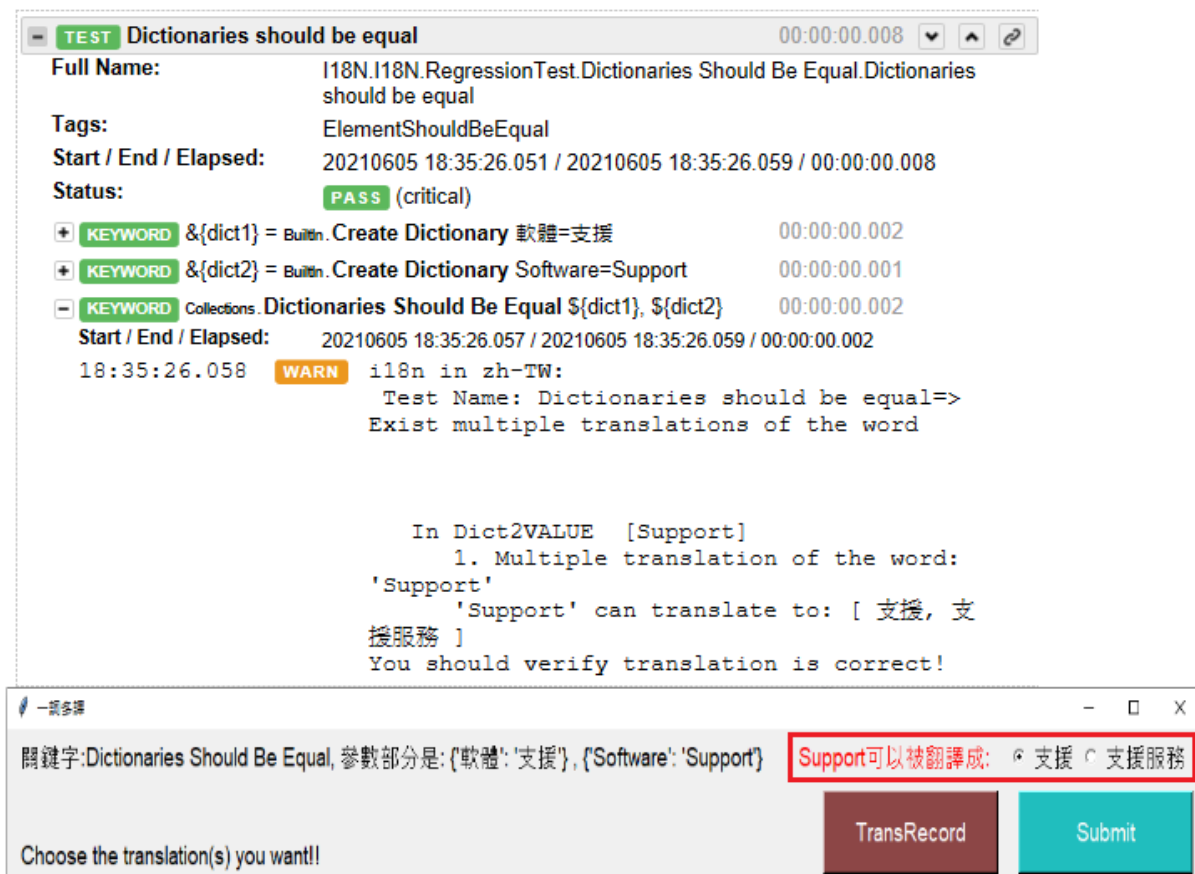


圖 4.8 Dictionaries Should Be Equal 測試腳本第一次執行通過

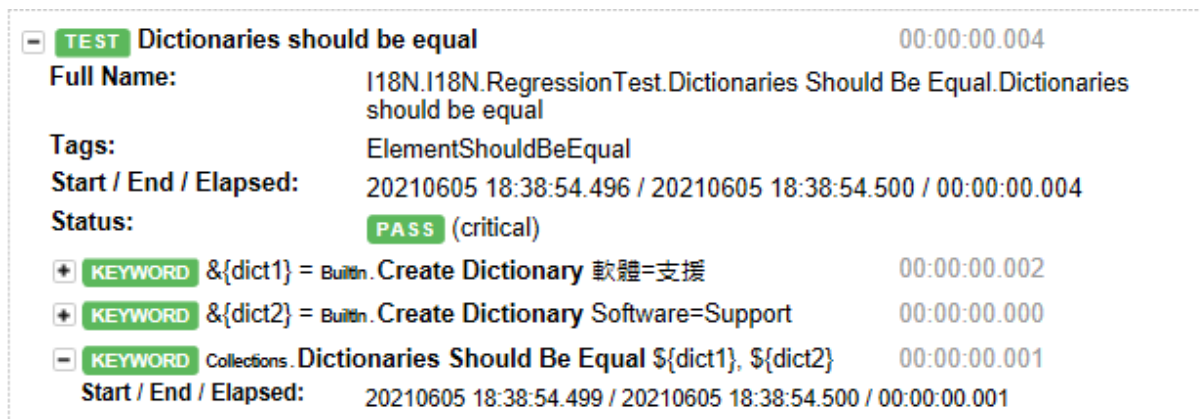


圖 4.9 Dictionaries Should Be Equal 測試腳本第二次執行通過

#### 4.1.4 Select From List By Label

Select From List By Label 的單元測試腳本 (如圖 4.10)，會打開 i18n Testing Website，並使用 Select From List By Label 選取畫面上 selection list 的 Support 選項 (如圖 4.11)。之後用 List Selection Should Be 驗證結果是否正確。第一次測試通過 (如圖 4.12)，而畫面上選擇了「支援」，代表參數 'Support' 有成功被翻譯，且系統將「支援」當作 'Support' 的其中一種翻譯，隨後跳出了一詞多譯 UI。之後，使用者選擇了「支援服務」當作 'Support' 的唯一翻譯。第二次測試腳本通過 (如圖 4.13)，畫面上被選擇的 selection list 顯示「支援服務」，代表參數 'Support' 成功被翻譯為使用者前一次的選擇。

```
*** Settings ***
Resource      ../CommonVariables.txt
Library       SeleniumLibrary
Test Setup    Run Keywords      Open Browser      http://localhost:3000
              Chrome
...           AND               Maximize Browser Window
Test Teardown Close Browser

*** Test Cases ***
Select from list by the label "Support"
${defaultSelection} = Set Variable    /*[@id='i18n-selection-list'
    ]/*[text()='Software' and @selected]
${selectionList} = Set Variable    /*[@id='i18n-selection-list']
Wait Until Element Is Visible    ${defaultSelection}    timeout=${
    shortPeriodOfTime}
Select From List By Label    ${selectionList}    Support
List Selection Should Be    ${selectionList}    Support
```

圖 4.10 Select From List By Label 在代理關鍵字下執行的測試腳本



TEST	Select from list by the label "Support"	00:00:03.687
Full Name: I18N.I18N.RegressionTest.Select From List By Label.Select from list by the label "Support"		
Tags: SelectFromListByLabel		
Start / End / Elapsed: 20210605 18:54:01.113 / 20210605 18:54:04.800 / 00:00:03.687		
Status: PASS (critical)		
+	SETUP BuiltIn.Run Keywords Open Browser, <a href="http://localhost:3000">http://localhost:3000</a> , Chrome, AND, Maximize Browser Window	00:00:01.337
+	KEYWORD \${defaultSelection} = BuiltIn.Set Variable /*[@id='i18n-selection-list']/*[text()='Software' and @selected]	00:00:00.000
+	KEYWORD \${selectionList} = BuiltIn.Set Variable /*[@id='i18n-selection-list']	00:00:00.001
+	KEYWORD SeleniumLibrary.Wait Until Element Is Visible \${defaultSelection}, timeout=\${shortPeriodOfTime}	00:00:00.024
-	KEYWORD SeleniumLibrary.Select From List By Label \${selectionList}, Support	00:00:00.104
Start / End / Elapsed: 20210605 18:54:02.477 / 20210605 18:54:02.581 / 00:00:00.104		
	18:54:02.537 INFO Selecting options from selection list '/*[@id='i18n-selection-list']' by label 支援服務.	
+	KEYWORD SeleniumLibrary.List Selection Should Be \${selectionList}, Support	00:00:00.097
+	TEARDOWN SeleniumLibrary.Close Browser	00:00:02.120

圖 4.13 Select From List By Label 測試腳本第二次執行通過





### 4.1.5 \*Page Should Contain Element

Page Should Contain Element 的單元測試腳本 (如圖 4.14)，會打開 Microsoft 中文官方網頁，前往支援頁面，之後使用 Page Should Contain Element 驗證畫面上的 'Support' 文字 (如圖 4.15)。第一次測試通過 (如圖 4.16)，代表參數 'Support' 有成功被翻譯，且系統將「支援」當作 'Support' 的其中一種翻譯，隨後跳出了一詞多譯 UI。之後，使用者選擇了「支援服務」當作 'Support' 的唯一翻譯。第二次測試腳本通過 (如圖 4.17)，且沒有跳出一詞多譯 UI，代表參數 'Support' 成功被翻譯為使用者前一次的選擇。

```
*** Settings ***
Resource    ../CommonVariables.txt
Library     SeleniumLibrary
Library     ../self_util.py
Test Setup  Run Keywords    Open Browser To Microsoft Page
...                               AND    Change Language    expectedLanguage=${
    language}
Test Teardown    Close Browser

*** Test Cases ***
Check "Microsoft Support" webelement is on the support page
Go To Support Page
${MicrosoftSupport} =    Set Variable    /*[@id = '
    supHomeAndLandingPageHeaderContainer']/*[contains(text(), '
    Support')]
Page Should Contain Element    ${MicrosoftSupport}
```

圖 4.14 \*Page Should Contain Element 在代理關鍵字下執行的測試腳本



圖 4.15 \*Page Should Contain Element 測試腳本要驗證的網頁元件 [22]

TEST

Check "Microsoft Support" webelement is on the support page

00:00:10.221

Full Name:

I18N.I18N.RegressionTest.Page Should Contain Element.Check "Microsoft Support" webelement is on the support page

Tags:

FindElement

Start / End / Elapsed:

20210605 19:11:04.573 / 20210605 19:11:14.794 / 00:00:10.221

Status:

PASS (critical)

SETUP

Button: Run Keywords Open Browser To Microsoft Page, AND, Change Language, expectedLanguage=\${language}

00:00:06.801

KEYWORD

Go To Support Page

00:00:01.285

KEYWORD

\$(MicrosoftSupport) = button.Set Variable //@id =supHomeAndLandingPageHeaderContainer//\*[contains(text(),'Support')]

00:00:00.001

KEYWORD

SeleniumLibrary.Page Should Contain Element \$(MicrosoftSupport)

00:00:00.020

Documentation:

Verifies that element locator is found on the current page.

Start / End / Elapsed:

20210605 19:11:12.661 / 20210605 19:11:12.681 / 00:00:00.020

KEYWORD

SeleniumLibrary.Get Webelement //@id =supHomeAndLandingPageHeaderContainer//\*[contains(text(),'支援')]

00:00:00.012

19:11:12.663

INFO

Detail Information

i18n in zh-TW :

Original Locator:

1. //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')]

Translated Locator:

1. //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), '支援')]

2. //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), '支援服務')]

19:11:12.663

WARN

i18n in zh-TW:

Test Name: Check "Microsoft Support" webelement is on the support page

locator: //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')]

In MULTI\_TRANS\_WORDS [Support]

1. Multiple translation of the word: 'Support'

'Support' can translate to: [ 支援, 支援服務 ]

You should verify translation is correct!

19:11:12.675

INFO

System use the locator: //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), '支援')]' to run!

19:11:12.681

INFO

Current page contains element '//\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')]'

TEARDOWN

SeleniumLibrary.Close Browser

00:00:02.113

關鍵字:Find Elements, 參數部分是: //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')]

Support可以被翻譯成: 支援服務 支援

Choose the translation(s) you want!!

TransRecord

Submit

圖 4.16 \*Page Should Contain Element 測試腳本第一次執行通過

[-] <b>TEST</b>	Check "Microsoft Support" webelement is on the support page	00:00:09.301
<b>Full Name:</b> I18N.I18N.RegressionTest.Page Should Contain Element.Check "Microsoft Support" webelement is on the support page		
<b>Tags:</b> FindElement		
<b>Start / End / Elapsed:</b> 20210605 19:18:40.363 / 20210605 19:18:49.664 / 00:00:09.301		
<b>Status:</b> <b>PASS</b> (critical)		
[+] <b>SETUP</b>	BuildIn.Run Keywords Open Browser To Microsoft Page, AND, Change Language, expectedLanguage=\${language}	00:00:05.879
[+] <b>KEYWORD</b>	Go To Support Page	00:00:01.285
[+] <b>KEYWORD</b>	\${MicrosoftSupport} = BuildIn.Set Variable <code>//*[@id='supHomeAndLandingPageHeaderContainer']//*[contains(text(),'Support')]</code>	00:00:00.001
[-] <b>KEYWORD</b>	SeleniumLibrary.Page Should Contain Element \${MicrosoftSupport}	00:00:00.010
<b>Documentation:</b> Verifies that element <code>locator</code> is found on the current page.		
<b>Start / End / Elapsed:</b> 20210605 19:18:47.530 / 20210605 19:18:47.540 / 00:00:00.010		
19:18:47.531	<b>INFO</b> Detail Information i18n in zh-TW : Original Locator: 1. <code>//*[@id='supHomeAndLandingPageHeaderContainer']//*[contains(text(),'Support')]</code> Translated Locator: 1. <code>//*[@id='supHomeAndLandingPageHeaderContainer']//*[contains(text(),'支援服務')]</code>	
19:18:47.540	<b>INFO</b> Current page contains element <code>//*[@id='supHomeAndLandingPageHeaderContainer']//*[contains(text(),'Support')]</code> .	
[+] <b>TEARDOWN</b>	SeleniumLibrary.Close Browser	00:00:02.123

圖 4.17 \*Page Should Contain Element 測試腳本第二次執行通過

#### 4.1.6 Table Should Contain

Table Should Contain 的單元測試腳本 (如圖 4.18)，會打開 I18n Testing Website，之後使用 Table Should Contain 驗證畫面上的 ‘Support’ 文字 (如圖 4.19)。第一次測試通過 (如圖 4.20)，代表參數 ‘Support’ 有成功被翻譯，且系統將「支援」當作 ‘Support’ 的其中一種翻譯，隨後跳出了一詞多譯 UI。之後，使用者選擇了「支援」當作 ‘Support’ 的唯一翻譯。第二次執行測試腳本通過 (如圖 4.21)，且沒有跳出一詞多譯 UI，代表參數 ‘Support’ 成功被翻譯為使用者前一次的選擇。

```
*** Settings ***
Resource    ../CommonVariables.txt
Library     SeleniumLibrary
Library     ../self_util.py
Test Setup  Run Keywords    Open Browser    http://localhost:3000
            Chrome
...                AND    Maximize Browser Window
Test Teardown    Close Browser

*** Test Cases ***
Table should contain "Support"
    ${table} =    Set Variable    /*[@id='i18n-table']
    Wait Until Element Is Visible    ${table}    timeout=${
        shortPeriodOfTime}
    Table Should Contain    ${table}    Support
```

圖 4.18 Table Should Contain 在代理關鍵字下執行的測試腳本

## I18n Testing Website

### alert()

For displaying the alert message, double click the "Show Alert Message" button:

Show Alert Message

### Selection List

軟體

### Table

Support對應到中文網頁上的翻譯

軟體	娛樂	支援
----	----	----

圖 4.19 Table Should Contain 測試腳本要驗證的網頁元件

TEST Table should contain "Support" 00:00:03.614

Full Name: I18N.I18N.RegressionTest.Table Should Contain.Table should contain "Support"

Tags: TableShouldContain

Start / End / Elapsed: 20210605 19:47:32.004 / 20210605 19:47:35.618 / 00:00:03.614

Status: PASS (critical)

SETUP BuiltIn.Run Keywords Open Browser, http://localhost:3000, Chrome, AND, Maximize Browser Window 00:00:01.335

KEYWORD \${table} = BuiltIn.Set Variable /\*[@id='i18n-table'] 00:00:00.000

KEYWORD SeleniumLibrary.Wait Until Element Is Visible \${table}, timeout=\${shortPeriodOfTime} 00:00:00.025

KEYWORD SeleniumLibrary.Table Should Contain \${table}, Support 00:00:00.139

Start / End / Elapsed: 20210605 19:47:33.366 / 20210605 19:47:33.505 / 00:00:00.139

KEYWORD SeleniumLibrary.Get WebElement /\*[@id='i18n-table'] 00:00:00.006

19:47:33.367 WARN i18n in zh-TW:  
Test Name: Table should contain "Support"=> Exist multiple translations of the word  
  
EXPECTED Word: 'Support'  
1. Multiple translations of the word: 'Support'  
'Support' can translate to: [ 支援服務, 支援 ]  
  
You should verify translation is correct!

TEARDOWN SeleniumLibrary.Close Browser 00:00:02.111

一語多譯

關鍵字:Table Should Contain, 參數部分是: /\*[@id='i18n-table'], Support

Support可以被翻譯成: ☒ 支援 ☐ 支援服務

Choose the translation(s) you want!!

TransRecord

Submit

圖 4.20 Table Should Contain 測試腳本第一次執行通過

50

[-] <b>SUITE</b> Table Should Contain	00:00:03.586
Full Name:	I18N.I18N.ReggressionTest.Table Should Contain
Source:	C:\Users\petje\OneDrive\桌面\thesis\i18n\i18n\RegressionTest\Table Should Contain.robot
Start / End / Elapsed:	20210605 19:53:49.318 / 20210605 19:53:52.904 / 00:00:03.586
Status:	1 critical test, 1 passed, 0 failed 1 test total, 1 passed, 0 failed
[-] <b>TEST</b> Table should contain "Support"	00:00:03.526
Full Name:	I18N.I18N.ReggressionTest.Table Should Contain.Table should contain "Support"
Tags:	TableShouldContain
Start / End / Elapsed:	20210605 19:53:49.375 / 20210605 19:53:52.901 / 00:00:03.526
Status:	<b>PASS</b> (critical)
[+] <b>SETUP</b> BuiltIn.Run Keywords Open Browser, http://localhost:3000, Chrome, AND, Maximize Browser Window	00:00:01.341
[+] <b>KEYWORD</b> \${table} = BuiltIn.Set Variable /*[@id='i18n-table']	00:00:00.001
[+] <b>KEYWORD</b> SeleniumLibrary.Wait Until Element Is Visible \${table}, timeout=\${shortPeriodOfTime}	00:00:00.023
[-] <b>KEYWORD</b> SeleniumLibrary.Table Should Contain \${table}, Support	00:00:00.032
Start / End / Elapsed:	20210605 19:53:50.741 / 20210605 19:53:50.773 / 00:00:00.032
[+] <b>TEARDOWN</b> SeleniumLibrary.Close Browser	00:00:02.126

圖 4.21 Table Should Contain 測試腳本第二次執行通過



## 4.2 改善翻譯邏輯後的驗收測試示例

以下將以 Page Should Contain Element 關鍵字的測試腳本為例 (如圖 4.22)，去驗證 Microsoft 中文官方網頁上的文字。並說明在改善 XPath 翻譯邏輯前後，執行含有 @placeholder 屬性的 XPath 之腳本，將分別得到何種結果。

此測試腳本會打開 Microsoft 中文官方網頁，並使用 Page Should Contain Element 驗證畫面上的 “How can we help you?” 文字。

```
*** Settings ***
Resource    ../CommonVariables.txt
Library     SeleniumLibrary
Library     ../self_util.py
Test Setup  Run Keywords    Open Browser To Microsoft Page
...                AND      Change Language    expectedLanguage=${
    language}
Test Teardown    Close Browser

*** Test Cases ***
Test web element is on the support page by given special attributes
Go To Support Page
    ${MicrosoftSupport} = Set Variable    //*[@id = '
        supHomeAndLandingPageSearchBox' and @placeholder = 'How can we
        help you?']
    Page Should Contain Element    ${MicrosoftSupport}
```

圖 4.22 含有 @placeholder 屬性的 XPath 之測試腳本

圖 4.23 則為網頁上含有 @placeholder 屬性的元件，且 @placeholder 屬性的值- “How can we help you”，對應到網頁上正確的翻譯應為 “我們該如何協助您?”

根據第一版和新版 i18n 工具的測試報表 (如圖 4.24、圖 4.25)，比對結果可得知: 第一版的 i18n 工具，會因為無法對 XPath 內的 @placeholder 屬性做翻譯，而使得測試發生錯誤。而新版的 i18n 工具，因為改善了翻譯邏輯，所以支援各種 HTML 屬性的翻譯。並成功將 “How can we help you?” 翻譯為「我們該如何協助您」，進而使測試通過。



圖 4.23 中文版 Microsoft 網頁 @placeholder 屬性對應的實際文字 [22]

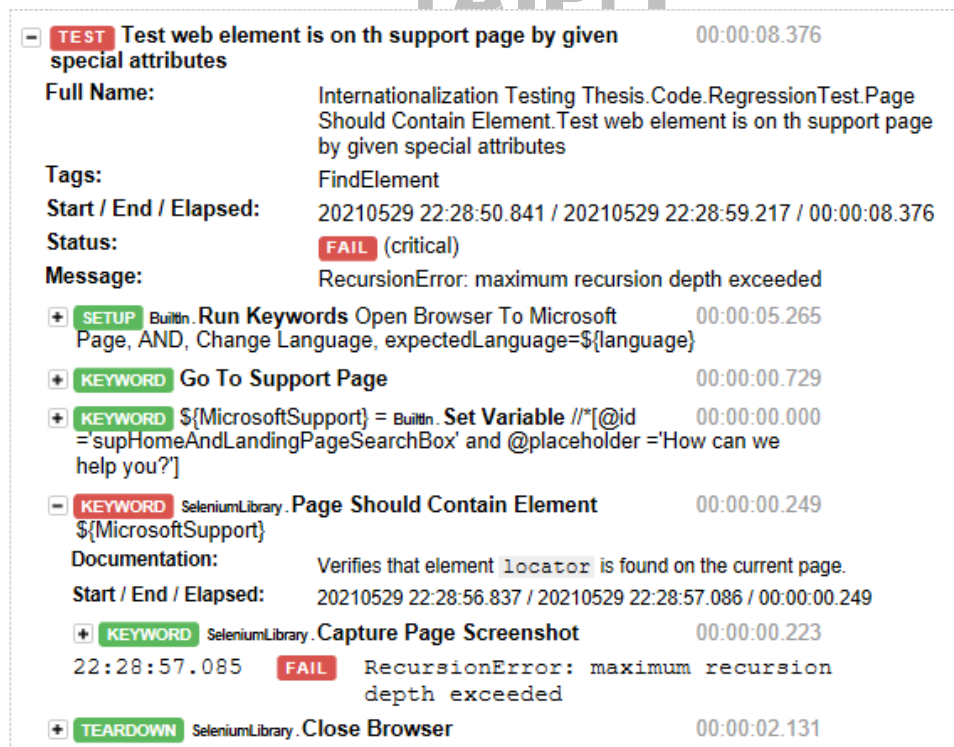


圖 4.24 待翻譯 XPath 運行在未改善翻譯邏輯時的 i18n 工具之結果



[-] <b>TEST</b>	Test web element is on th support page by given special attributes	00:00:09.312
<b>Full Name:</b> I18N.I18N.RegressionTest.Page Should Contain Element.Test web element is on th support page by given special attributes		
<b>Tags:</b> FindElement		
<b>Start / End / Elapsed:</b> 20210529 22:16:42.690 / 20210529 22:16:52.002 / 00:00:09.312		
<b>Status:</b> <b>PASS</b> (critical)		
[+] <b>SETUP</b>	Builtin. Run Keywords Open Browser To Microsoft Page, AND, Change Language, expectedLanguage=\${language}	00:00:05.639
[+] <b>KEYWORD</b>	Go To Support Page	00:00:01.533
[+] <b>KEYWORD</b>	\${MicrosoftSupport} = Builtin. Set Variable <code>//*[@id='supHomeAndLandingPageSearchBox' and @placeholder='How can we help you?']</code>	00:00:00.000
[-] <b>KEYWORD</b>	SeleniumLibrary. Page Should Contain Element <code>\${MicrosoftSupport}</code>	00:00:00.012
<b>Documentation:</b> Verifies that element <code>locator</code> is found on the current page.		
<b>Start / End / Elapsed:</b> 20210529 22:16:49.864 / 20210529 22:16:49.876 / 00:00:00.012		
22:16:49.866	<b>INFO</b> Detail Information i18n in zh-TW : Original Locator: 1. <code>//*[@id='supHomeAndLandingPageSearchBox' and @placeholder='How can we help you?']</code> Translated Locator: 1. <code>//*[@id='supHomeAndLandingPageSearchBox' and @placeholder='我們該如何協助您?']</code>	
22:16:49.875	<b>INFO</b> Current page contains element <code>'//*[@id='supHomeAndLandingPageSearchBox' and @placeholder='How can we help you?']'</code> .	
[+] <b>TEARDOWN</b>	SeleniumLibrary. Close Browser	00:00:02.126

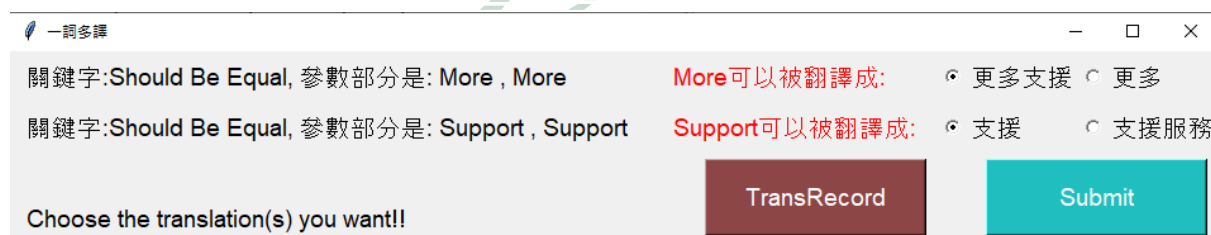
圖 4.25 待翻譯 XPath 運行在改善翻譯邏輯後的 i18n 工具之結果

### 4.3 一詞多譯情況下所產生的圖形化使用者介面

在遭遇一詞多譯且測試腳本會通過的情況下，以 Test should be equal 為例 (如圖 4.26)；分別創出兩個變數 word1、word2，並用 Should Be Equal 檢驗兩個變數的值是否正確。測試執行結束後，系統會產生一個圖形化使用者介面 (如 4.27)，上面分別記錄了遭遇一詞多譯的關鍵字完整參數，以及所有可能的翻譯詞，供使用者做選擇。

```
*** Settings ***
Test should be equal
    ${word1} =      Set Variable      More
    ${word2} =      Set Variable      Support
    Should Be Equal    ${word1}      More
    Should Be Equal    ${word2}      Support
```

圖 4.26 Test should be equal 測試腳本



一詞多譯

關鍵字:Should Be Equal, 參數部分是: More , More      More可以被翻譯成:    ☒ 更多支援    ☐ 更多

關鍵字:Should Be Equal, 參數部分是: Support , Support      Support可以被翻譯成:    ☒ 支援    ☐ 支援服務

Choose the translation(s) you want!!    TransRecord    Submit

圖 4.27 遭遇一詞多譯且測試通過，開啟一詞多譯 UI

當使用者選擇了希望的翻譯並按下 Submit 按鈕後，系統便會產生一份設定檔，以利下次執行到同一份腳本的同關鍵字時，直接套用選擇的翻譯，同時清空一詞多譯介面上的翻譯選項 (如圖 4.28)。

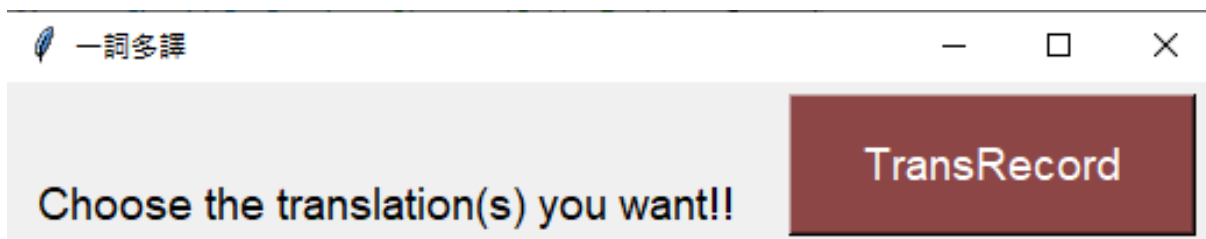


圖 4.28 按下 Submit 按鈕後，將翻譯寫進設定檔，同時清除翻譯選項

假如使用者選擇錯誤時，則可以點擊 TransRecord 按鈕來打開翻譯紀錄的介面，上面記錄著使用者曾經的翻譯選擇，並有一顆 Undo 按鈕 (如圖 4.29)。使用者選擇了要清除的翻譯紀錄後，按下 Undo 按鈕，系統便會將設定檔中的該資料刪除，同時關閉翻譯紀錄介面，當再次打開翻譯紀錄介面時，可以發現被選擇清除的資料已消失 (如圖 4.30)。



圖 4.29 打開翻譯紀錄介面，上面記錄著使用者翻譯選擇

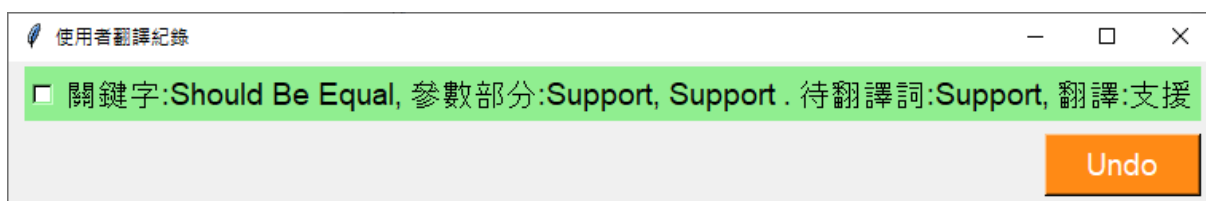


圖 4.30 清除選擇的翻譯紀錄後，再次打開翻譯紀錄介面

然而，並非所有的測試腳本都會遭遇一詞多譯，且總是通過。因此，底下將分別說明在其他兩種情況下，系統會有甚麼不同的反應：

1. 測試腳本不通過: 若測試腳本不通過，則不論遭遇一詞多譯與否，皆不會跳出一詞多譯 UI。其設計想法為，因為系統有檢驗的機制，所以此刻測試腳本不會通過，則代表原先未翻譯前的測試腳本也不會通過。而為原先就會發生錯誤的待翻譯詞選擇其翻譯，對使用者而言，顯然是毫無意義的，但是為了測試結果的完整性，系統還是會於測試報表上顯示一詞多譯的 warning 資訊(如圖 4.31)。

The screenshot displays a test report for a test case named "Test should be equal". The test status is "FAIL (critical)". The message indicates a failure: "更多 != 支援服務". A warning message is also present, stating: "WARN i18n in zh-TW: Test Name: Test should be equal=> Exist multiple translations of the word. First Word: 'More'. 1. Multiple translations of the word: 'More'. 'More' can translate to: [ 更多, 更多支援 ]. Second Word: 'Support'. 1. Multiple translations of the word: 'Support'. 'Support' can translate to: [ 支援服務, 支援 ]. You should verify translation is correct!". The warning message is highlighted with a red box. The test report also shows the full name, tags, start/end/elapsed times, and a final failure message: "FAIL 更多 != 支援服務".

圖 4.31 測試腳本不通過的情況下，不跳出 UI，但會顯示 warning 資訊於報表

2. 測試腳本通過，且沒有遭遇一詞多譯: 在測試腳本通過且沒有遭遇一詞多譯的情況下，測試結束後便不會跳出一詞多譯 UI 介面。但有一種情況是，關鍵字的參數部分原本會遭遇一詞多譯，但因為前一次執行時使用者已針對此待翻譯詞選擇了翻譯，所以被系統視為只有一種翻譯，因此不跳出一詞多譯 UI(如圖 4.32)。(有關係統如何判定翻譯已被使用者選擇，詳見本論文 3-2 節)

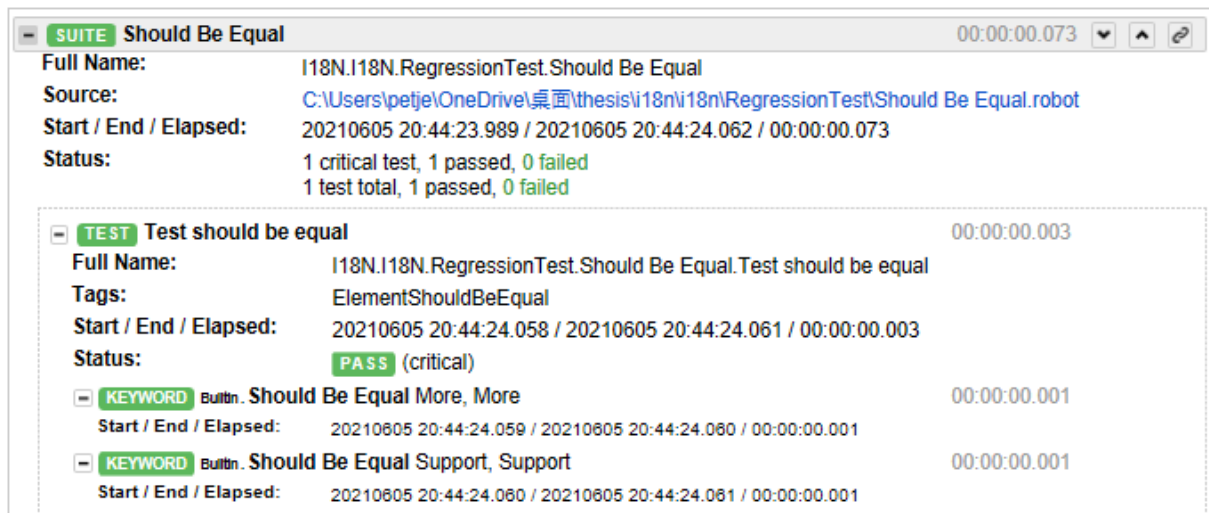


圖 4.32 測試腳本通過且沒有遭遇一詞多譯的情況下，不跳出 UI



## 4.4 使用涵蓋多項代理關鍵字的測試腳本

以上介紹的測試案例，分別呈現了本論文各項功能改善後的實際成果，但因為都偏向單一關鍵字或單一功能的測試，在此將設計一個較複雜的 Robot Framework 網頁自動化驗收測試腳本，內部使用到多種代理關鍵字，來對 Microsoft 的中文版網頁進行一系列操作。同時，會遭遇到一詞多譯，並且會使用相別於第一版 i18n 所列舉的 HTML 屬性來撰寫 XPath。藉此來對新版 i18n 工具作一個驗收測試。

以下將執行 “Test multiple user behaviors on Microsoft website” 測試腳本兩次，分別呈現選擇一詞多譯翻譯前後，測試結果有何差異之處。



此測試腳本 (如圖 4.33)，會打開 Microsoft 中文官方網頁，先到支援頁面，之後點擊「支援」的 Logo，到新的支援頁面。最後，驗證畫面上的橫向標語有包含 'Support' 字樣，以及畫面上搜尋框有包含 "How can we help you" 字樣。因為同時遭遇到 3 種 'Support' 需要被翻譯，系統便會逐一將它們紀錄下來，待測試腳本結束後，分別顯示於一詞多譯 UI 上。

```
*** Settings ***
Resource      ./Keywords.txt
Library       SeleniumLibrary
Test Setup    Run Keywords      Open Browser To Microsoft Page
...           AND              Change Language      expectedLanguage=${
    language}
Test Teardown  Close Browser

*** Test Cases ***
Test multiple user behaviors on Microsoft website
    Go To Support Page
    Support Button Text Should Be      Support
    Wait Until Element Is Visible     /*[@id = 'uhfCatLogo' ]/*[
        normalize-space()='Support']
    Click Element                      /*[@id = 'uhfCatLogo' ]/*[normalize-space()='
        Support']
    ${MicrosoftSupport} = Set Variable /*[@id = '
        supHomeAndLandingPageHeaderContainer']/*[contains(text(), '
        Support')]
    ${searchBox} = Set Variable /*[@id = '
        supHomeAndLandingPageSearchBox' and @placeholder ='How can we
        help you?']
    Page Should Contain Element        ${MicrosoftSupport}
    Page Should Contain Element        ${searchBox}

*** Keywords ***
Support Button Text Should Be
    [Arguments]    ${expected}
    ${supportButton} = Set Variable /*[@id = 'uhfCatLogo']
    Element Text Should Be    ${supportButton}    ${expected}
```

圖 4.33 Test multiple user behaviors on Microsoft website 測試腳本實作

第一次執行測試腳本(如圖 4.34)結束後，使用者分別為三種不同情況下遭遇一詞多譯的‘Support’，選擇了「支援」、「支援」、「支援服務」為其正確翻譯，並點擊 Submit 按鈕，將選擇寫入設定檔中。

The screenshot displays a Selenium test execution log for a test titled "Test multiple user behaviors on Microsoft website". The log shows various steps including "Run Keywords", "Go To Support Page", and "Page Should Contain Element". A warning message at the bottom of the log indicates a translation issue with the word "Support".

Below the log, a translation selection interface is shown. It lists three instances of the word "Support" and provides three translation options for each: "支援服務", "支援", and "支援服務". The user has selected "支援服務" for all three instances. The interface includes a "Submit" button and a "TransRecord" button.

TEST Test multiple user behaviors on Microsoft website 00:00:11.278

Full Name: 118N.118N.ReggressionTest.Multiple User Behavior.Test multiple user behaviors on Mi website

Tags: ElementTextShouldBe

Start / End / Elapsed: 20210616 16:37:08.825 / 20210616 16:37:20.103 / 00:00:11.278

Status: PASS (critical)

SETUP BuiltIn.Run Keywords Open Browser To Microsoft Page, AND, Change Language, expectedLanguage=\${language} 00:00:06.875

KEYWORD Keywords.Go To Support Page 00:00:01.190

KEYWORD Support Button Text Should Be Support 00:00:00.038

KEYWORD SeleniumLibrary.Wait Until Element Is Visible //\*[@id = 'uhfCatLogo']//\*[normalize-space()='Support'] 00:00:00.029

KEYWORD SeleniumLibrary.Click Element //\*[@id = 'uhfCatLogo']//\*[normalize-space()='Support'] 00:00:00.935

KEYWORD \${MicrosoftSupport} = BuiltIn.Set Variable //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')] 00:00:00.001

KEYWORD \${searchBox} = BuiltIn.Set Variable //\*[@id = 'supHomeAndLandingPageSearchBox'] and @placeholder = 'How can we help you?' 00:00:00.001

KEYWORD SeleniumLibrary.Page Should Contain Element \${MicrosoftSupport} 00:00:00.036

Documentation: Verifies that element locator is found on the current page.

Start / End / Elapsed: 20210616 16:37:17.898 / 20210616 16:37:17.934 / 00:00:00.036

KEYWORD SeleniumLibrary.Get Webelement //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), '支援')] 00:00:00.027

16:37:17.899 INFO Detail Information

118n in zh-TW :

Original Locator:

1. //\*[@id = 'supHomeAndLandingPageHeaderContainer'] [contains(text(), 'Support')]

Translated Locator:

1. //\*[@id = 'supHomeAndLandingPageHeaderContainer'] [contains(text(), '支援')]

2. //\*[@id = 'supHomeAndLandingPageHeaderContainer'] [contains(text(), '支援服務')]

16:37:17.900 WARN 118n in zh-TW:

Test Name: Test multiple user behaviors on Microsoft website

locator: //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(t(), 'Support')]

In MULTI\_TRANS\_WORDS [Support]

1. Multiple translation of the word: 'Support'

'Support' can translate to: [ 支援, 支援服務 ]

You should verify translation is correct!

關鍵字:Element Text Should Be, 參數部分是: //\*[@id = 'uhfCatLogo'], Support

關鍵字:Find Elements, 參數部分是: //\*[@id = 'uhfCatLogo']//\*[normalize-space()='Support']

關鍵字:Find Elements, 參數部分是: //\*[@id = 'supHomeAndLandingPageHeaderContainer']//\*[contains(text(), 'Support')]

Support可以被翻譯成: ☐ 支援服務 ☒ 支援 ☐ 支援

Support可以被翻譯成: ☐ 支援服務 ☒ 支援 ☐ 支援

Support可以被翻譯成: ☐ 支援服務 ☒ 支援 ☐ 支援

Choose the translation(s) you want!!

TransRecord Submit

圖 4.34 Test multiple user behaviors on Microsoft website 測試腳本第一次執行結果



第二次執行測試腳本時(如圖 4.35)，會根據設定檔內的翻譯紀錄，分別對三種‘Support’進行翻譯。因為‘Support’被 i18n 系統根據不同情況判定為含有唯一翻譯，所以測試結束後，並不會顯示一詞多譯 UI。所以測試結束後，並不會顯示一詞多譯 UI。

```
- TEST Test multiple user behaviors on Microsoft website 00:00:08.972
Full Name: I18N.I18N.RegressionTest.Multiple User Behavior.Test multiple user behaviors on Microsoft website
Start / End / Elapsed: 20210623 09:33:52.990 / 20210623 09:34:01.962 / 00:00:08.972
Status: PASS (critical)

+ SETUP BuiltIn.Run Keywords Open Browser To Microsoft Page, AND, Change Language, expectedLanguage=${language} 00:00:04.757
+ KEYWORD Keywords.Go To Support Page 00:00:01.113
- KEYWORD Support Button Text Should Be Support 00:00:00.028
Start / End / Elapsed: 20210623 09:33:58.861 / 20210623 09:33:58.889 / 00:00:00.028
+ KEYWORD ${supportButton} = BuiltIn.Set Variable /*[@id = 'uhfCatLogo'] 00:00:00.000
- KEYWORD SeleniumLibrary.Element Text Should Be ${supportButton}, ${expected} 00:00:00.027
Start / End / Elapsed: 20210623 09:33:58.862 / 20210623 09:33:58.889 / 00:00:00.027
09:33:58.863 INFO 'Support' is currently resolved as '支援'
09:33:58.863 INFO Verifying element '//*[@id = 'uhfCatLogo']' contains exact text '支援'.
- KEYWORD SeleniumLibrary.Wait Until Element Is Visible /*[@id = 'uhfCatLogo']/*[normalize-space()='Support'], 00:00:00.021
timeout=${shortPeriodOfTime}
Documentation: Waits until the element locator is visible.
Start / End / Elapsed: 20210623 09:33:58.889 / 20210623 09:33:58.910 / 00:00:00.021
09:33:58.890 INFO Detail Information
i18n in zh-TW :
Original Locator:
1. /*[@id = 'uhfCatLogo'] /*[normalize-space()='Support']
Translated Locator:
1. /*[@id = 'uhfCatLogo'] /*[normalize-space()='支援']

+ KEYWORD SeleniumLibrary.Click Element /*[@id = 'uhfCatLogo']/*[normalize-space()='Support'] 00:00:00.901
+ KEYWORD ${MicrosoftSupport} = BuiltIn.Set Variable /*[@id = 'supHomeAndLandingPageHeaderContainer']/* 00:00:00.000
[contains(text(), 'Support')]
+ KEYWORD ${searchBox} = BuiltIn.Set Variable /*[@id = 'supHomeAndLandingPageSearchBox' and @placeholder = 'How 00:00:00.000
can we help you?']
- KEYWORD SeleniumLibrary.Page Should Contain Element ${MicrosoftSupport} 00:00:00.015
Documentation: Verifies that element locator is found on the current page.
Start / End / Elapsed: 20210623 09:33:59.813 / 20210623 09:33:59.828 / 00:00:00.015
09:33:59.815 INFO Detail Information
i18n in zh-TW :
Original Locator:
1. /*[@id = 'supHomeAndLandingPageHeaderContainer']/*[contains(text(), 'Support')]
Translated Locator:
1. /*[@id = 'supHomeAndLandingPageHeaderContainer']/*[contains(text(), '支援服務')]
09:33:59.828 INFO Current page contains element '//*[@id = 'supHomeAndLandingPageHeaderContainer']/*[contains(text(),
'Support')]'.
+ KEYWORD SeleniumLibrary.Page Should Contain Element ${searchBox} 00:00:00.012
+ TEARDOWN SeleniumLibrary.Close Browser 00:00:02.122
```

圖 4.35 Test multiple user behaviors on Microsoft website 測試腳本第二次執行結果

## 第五章 結論與未來展望

### 5.1 結論

本論文提出了四項措施去改進第一版 i18n 工具所面臨的問題，分別是擴充代理關鍵字、支援所有 HTML 屬性的 XPath 翻譯邏輯、支援自由選擇翻譯的一詞多譯圖形化使用者介面、可以提供安裝的 Python 模組。透過以上努力，即可讓使用此 i18n 工具的多國語言網頁自動化驗收測試腳本，不再侷限於只能使用特定的關鍵字去撰寫。並且 XPath 的撰寫也不再需要考慮只有 @title、text()、normalize-space() 會被翻譯的問題，使用者的測試腳本可以用更多元的 XPath 寫法來搭配此 i18n 工具使用。遭遇一詞多譯時，使用者也能夠在第一次測試腳本結束後，選擇內心所預期的翻譯詞來執行接下來的測試。最後，藉由將 i18n 工具包裝成可以 pip install 的 Python 模組，使用者可以隨安裝即使用本工具所提供的功能，而無須將自己的測試腳本移動到 i18n 工具的資料夾下。

## 5.2 i18n 工具的使用限制與未來展望

本論文的 i18n 工具尚存在著使用限制，其原因之一源自於某些 Robot Framework 的原生關鍵字，其參數部分接受的是特殊的字元，目前 i18n 工具是不支援的。如圖 5.1，Get Match Count 接受的第二個參數是 'S\*'，用來抓出串列中符合的字串，若是強行將 'S\*' 翻譯，便會直接導致測試結果出錯。

```
*** Test Cases ***
Get match count
    @list1 =      Create List      Support      Software      Apple
    ${count} =    Get Match Count    ${list1}      S*
    Log          ${count}
```

圖 5.1 Get Match Count 測試腳本

此外，某些 Robot Framework 的關鍵字，其內部實作會呼叫另一個原生的關鍵字，且被呼叫的關鍵字也有一詞多譯的需求。當使用者為第一次執行測試的一詞多譯選擇了翻譯後，並執行第二次測試腳本時，因為此時傳入代理關鍵字的參數部分已改變，系統會將其判定為不同的待翻譯詞，而需要使用者事後再選擇一次。如圖 5.2，Dictionary Should Contain Item 原生關鍵字的實作會呼叫 Dictionary Should Contain Key(如圖 5.3 紅色框起部分)，導致第一次執行測試時傳入的 'More'，和第二次執行時傳入的 'More'，由於參數紀錄的局部從 [ '支援'，'支援服務' ] 變成 [ '支援' ]，因此被系統判定成不同的待翻譯詞，需要重新選擇(如圖 5.4)。

```
*** Test Cases ***
Dictionary should contain item
    &dict1 =      Create Dictionary      More=Support
    Dictionary Should Contain Item      ${dict1}      More      Support
```

圖 5.2 Dictionary Should Contain Item 測試腳本

```
def dictionary_should_contain_item(self, dictionary, key, value, msg=None):
    """An item of ``key`` / ``value`` must be found in a ``dictionary``.

    Value is converted to unicode for comparison.

    Use the ``msg`` argument to override the default error message.
    """
    self.validate_dictionary(dictionary)
    self.dictionary_should_contain_key(dictionary, key, msg)
    actual, expected = unic(dictionary[key]), unic(value)
    default = "Value of dictionary key '%s' does not match: %s != %s" % (key, actual, expected)
    _verify_condition(actual == expected, default, msg)
```

圖 5.3 Dictioanry Should Contain Item 原生實作



圖 5.4 第一次與第二次執行測試腳本，待翻譯詞的參數紀錄變化

未來展望的部分，可以根據 i18n 工具尚存在的使用限制，未來朝向優化特殊字元的翻譯方式，以及改善重複翻譯詞的判定邏輯去努力。而且目前 JSON 翻譯檔還需要由使用者提供，未來可以研究如何透過設計爬蟲，抓取多國語言網頁上的對應翻譯詞，自動將 JSON 翻譯檔產生出來，將人工的成本降低。進而讓「多國語言網頁自動化驗收測試」工具變得更貼近測試者的需要，錯誤率低且容易使用。

## 參考文獻

- [1] 朱峻平. 支援多國語言的 Robot Framework 網頁自動化驗收測試. Master's thesis, 國立臺北科技大學資訊工程系碩士班, 臺北, 2019.
- [2] Lindsay Bassett. *Introduction to JavaScript object notation: a to-the-point guide to JSON*. "O'Reilly Media, Inc.", 2015.
- [3] Jane Knight. Updated definition of internationalization. *International higher education*, (33), 2003.
- [4] Robot Framework Official Website. <https://robotframework.org/>. Online; accessed 11 June 2021.
- [5] James Clark, Steve DeRose, et al. XML path language (XPath), 1999.
- [6] 呂昭陞. 一個改善 XPath 表達式在網頁應用程式自動化驗收測試之穩定性的方法. Master's thesis, 國立臺北科技大學資訊工程系碩士班, 臺北, 2018.
- [7] 李允中. 軟體工程, page 194. 臺北, 社團法人臺灣軟體工程學會, 2013.
- [8] Chin-Yun Hsieh, Chen-Hsin Tsai, Yu Chin Cheng. Test-Duo: A framework for generating and executing automated acceptance tests from use cases. In *8th International Workshop on Automation of Software Test (AST)*, San Francisco, CA, USA, 2013, May.
- [9] PIP Project Description. <https://pypi.org/project/pip/>. Online; accessed 11 June 2021.

- [10] Rigzin Angmo and Monika Sharma. Performance evaluation of web based automation testing tools. In *2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence)*, pages 731–735. IEEE, 2014.
- [11] AngularJS Official Website. <https://angularjs.org/>. Online; accessed 27 June 2021.
- [12] angular-translate Official Website. <https://angular-translate.github.io/>. Online; accessed 27 June 2021.
- [13] Robot Framework User Guide. <http://robotframework.org/robotframework/3.2.2/RobotFrameworkUserGuide.html>. Online; accessed 11 June 2021.
- [14] Python Official Website. <https://www.python.org/>. Online; accessed 11 June 2021.
- [15] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, pages 207–217. Addison-Wesley, 1994.
- [16] RED – Robot Editor. <https://github.com/nokia/RED/>. Online; accessed 11 June 2021.
- [17] Robot Framework Official Github Website. <https://github.com/robotframework/robotframework>. Online; accessed 27 June 2021.
- [18] SeleniumLibrary Official Github Website. <https://github.com/robotframework/SeleniumLibrary>. Online; accessed 27 June 2021.
- [19] PYPI Official Website. <https://pypi.org/>. Online; accessed 11 June 2021.
- [20] Michael L Katz and Carl Shapiro. How to license intangible property. *The Quarterly Journal of Economics*, 101(3):567–589, 1986.

[21] Per Runeson. A survey of unit testing practices. *IEEE software*, 23(4):22–29, 2006.

[22] Microsoft Official Website. <https://www.microsoft.com/>. Online; accessed 11 June 2021.

[23] Node.js Official Website. <https://nodejs.org/en/>. Online; accessed 11 June 2021.

