工業4.0機台分析開發日誌

# 2017.07.05 (三)

我們與 陳石坤博士與 丁國正老師於上午10點開會，決定在暑期兩個月內完成**兩項專案**，一項是關於工業4.0的機台分析，透過機台吐出的資料與生管處所排出的工單資料做比對，以視覺化的方式呈現實際情況，並以智慧化的排程提供給生管處做為參考，提升生管處與生產處的連結性，進而增加工廠產能，另一項則是AGV無人搬運車的規劃，由我們三人共同完成這兩項專案，並詳細的將**工作分配**，前者由蔡昌銘主導，後者由劉邵宸主導，陳柏翔則在兩邊互相支援，而本文件將詳述機台分析的開發日誌，AGV將撰寫於另一份文件中，因此不在此贅述。

# 2017.07.06 (四)

此Project先前已有學長研究，並且在畢業前交接給我們，但是我們利用學長留下來的程式碼執行新的資料(07/03)，卻會產生**例外狀況**，我們嘗試了一整天都無法排除，於是我們討論決定，改用我們較為熟悉的**Python**作為主要開發的程式語言，並全部重新編寫。

# 2017.07.07 (五)

今天我們開始處理資料，透過學長的經驗，我們知道程式代碼是我們最主要的資訊，但是該如何處理這些資料我們卻沒有什麼概念，因此切割完之後，我們花了很多時間在**思考資料結構的設計**上面，我們首先想到的方式是將切割好的資料以Dictionary作為分類的依據，再以List包起來，但是資料量類型相當的多，處理起來有點困難。

# 2017.07.10 (一)

我們嘗試利用上禮拜設計的資料結構進行繪製折線圖，由於資料層級太多的關係，呼叫起來非常困難，所以做到一半我們便決定**重新設計資料儲存**的方式，以雙層**Class**的方式格式化資料，透過Object的建立將不同的資料以相同的格式儲存起來，如此一來，就算資料的類型再多，也能很輕鬆的存取。

# 2017.07.11 (二)

為了符合生管處的需求，我們在視覺化之前要先將Object裡的資料轉換成線性的資料，我們的想法是先宣告一個擁有86400個空間的list，因為一天中有86400秒，之後給當天每個使用到的Program附上一個流水號，再透過這個list一個一個將Object裡的資料對應出來，然後再Import一個叫做matplotlib的插件，呼叫此插件內建的method將list畫成折線圖。

# 2017.07.12 (三)

今天上午我們在做下午開會所需的簡報，下午我們與 陳石坤博士以及 葉家僖博士開會時，提出了幾個問題：

1. 工單與機台資料**無法對應**，就算以學長之前的方法也失敗
   * 由於DMG進度超前，因此擴大工單尋找範圍應該可以找到
2. DMG#1已經做得差不多了，可以開始**著手其他機台**
   * 再來會提供DMG#2的資料，可以將兩個機台的折線圖合併
3. 除了每天的折線圖以外，是否還有需要其他的呈現方式
   * 將每個**零件的製作時間**統計成Excel表格，並計算稼動率

# 2017.07.13 (四)

【公假：參訪清泉崗廠區】

# 2017.07.14 (五)

由於我們之前都是以一個機台的概念下去設計程式碼，繪圖的Class無法導入多個機台的資料進行合併繪圖，因此我們**重新組織**了整個專案的Package結構，我們將分割資料與格式化儲存歸為一類，線性化資料繪製折線圖歸類為另一類。其中，原先線性化資料與繪製折線圖是一口氣做完的設計方式，為了因應多機台的將**線性化獨立**出來，全部線性化完成後，才開始做繪圖的動作。

# 2017.07.17 (一)

今天我們開始著手匯出Excel統計表藉由Import一個叫做openpyxl的插件，產出時間模型，是生管處重點需求之一，但是在我們剛開始沒多久就發現一個嚴重的問題，生管處開給我們的欄位中，有幾個欄位只有工單裡面才找得到，因此我們必須在工單與機台資料間找到可靠的Key值，以方便我們補齊機台資料缺少的欄位，但是一時之間我們無法找到哪些欄位足夠可靠，所以我們決定先將能在機台資料內找到的欄位建成時間模型。

# 2017.07.18 (二)

由於折線圖的需要，格式化的資料是以天作為單位，但是時間模型的最小單位是件號，因此我們必須把有跨天製作的資料合併起來，而其中的WARM與TOOL-SL-D.I兩種狀況我們無法判斷該怎麼合併，於是我們找了 葉家僖博士來釐清這個問題，初步判斷WARM是暖機狀態，應該是不用納入時間模型內計算，也包含暖機前開機的那一小段程式狀態，而TOOL-SL-D.I是刀具定位的程式，按常理來說，應該是要與前後程式合併計算，之後與生管處開會將再次確認這樣判斷是否正確。

# 2017.07.19 (三) 去工廠討論Key

在機台資料包含的欄位都建成時間模型後，我們需要補上工單才找得到的欄位，我們用人工的方式卻找不到任何一個欄位是可靠的Key值，總是會有一兩筆資料會呈現一對多的情況，於是葉家僖博士帶我們到工廠找領班詢問，領班建議我們利用工具代碼往回追朔版別、NC件號與零件編號，因為NC件號與零件編號是一對一，版別跟工具代碼也是一對一，可是前兩個與後兩個卻是一對多，所以往回推才不會找到多個結果，但是我們回來再次確認還是發現有重複的情況，所以我們又去了第二次工廠，領班補充說：零件編號前面有TS代表TEST是測試用的，建議排除這個排程，而版別若與工具代碼沒有呈現一對一的情況，則是該程式有經歷小幅度的改動，因此代碼有更新而版別卻沒有更新，綜合上述的情況，我們決定以零件編號與工具代碼作為我們的雙重Key值，以Double check的方式去搜尋工單內我們需要的欄位。另外我們發現工號呈現一對多的情況，在工單與實際機台運作狀況落差過大時，很難去對應出正確的工號，這個問題我們也暫時擱置，直到未來與生管處開會再提出來釐清

# 2017.07.20 (四)

DMG01\_20170708\_20170716資料進來，發現判斷式的漏洞，禮拜一機台開機時需要運作一下，才能做暖機的動作(WARM\_UP1.H)，而這運作一下的資料卻是上禮拜所做的東西，造成加工工時多算的bug，解決方式不能只判斷隔一天就把兩筆相同工具代碼合併，要多判斷有沒有暖機的動作，把兩筆資料分開做

# 2017.07.21 (五)

整理文件與資料，此目的是避免失誤把機台資料與程式碼一併上傳至Github，當前的機台資料是不可以公開的，將資料與程式碼分別放進不同的資料夾，程式碼所產出的圖片與Excel也分開放置，讓往後找尋檔案之時能夠快速找到所需資料(圖片:graph,時間模型excel:time\_model,機台資料:2017-07)

# 2017.07.24 (一)

與陳石坤博士和丁國正老師，討論出這禮拜主要目的著重於，增加其他的機台的資料，做分析，主要擔心的是其他機台資料，格式不同需要重新切割資料，啟動狀態與停機狀態判斷式也有可能不一樣，有可能需要很大的修改程式碼，目前工號依然還是對不到

# 2017.07.25 (二)

今天STC機台的資料下來了，我們比對了一下STC與DMG的資料，發現除了少了一些欄位以外，其實差異性不大，若要替STC另外撰寫資料切割的程式碼，會使整個系統過於繁瑣，於是我們重新思考了整個系統的思路，以現有的程式碼重新組職一個新的專案，以適應輸入端有多種機台的情況，先前的方式是將機台資料與工單湊成一組一組的丟到系統裡，再依序將各機台的結果輸出，現在我們只需將所有檔案放到指定的資料夾內，系統便會自動將檔案做分類，分別交給不同class處理，就不需要再透過替檔案重新命名來配對機台資料與工單，且若配對不起來，系統也會自動略過不處理，不會再有例外發生。

# 2017.07.26 (三)

今天我們把資料切割與資料格式化重新組織完成了，改動的幅度沒有很大，算是蠻順利就完成了，雖然整個演算法的核心概念沒有改變，但是從機台資料與工單已經沒有包在同一個變數裡面了，因此後面的作圖需要做相較大幅度的修改，第一個需要解決的是機台資料與工單的配對，這個問題我們暫時擱置了，因為家熙博士剛好有空來關心我們的進度，所以我們與他討論時間模型的演算法，因為先前都只有DMG一種機台，我們可以將暖機狀態特別拉出來做處理，但是家熙博士建議我們不要這樣做，因為STC暖機狀態的代碼長得不一樣，這樣每新增一種機台就要多寫一種處理方式，非常累贅，我們最後討論出的結果是，先把正常的工具代碼辨識出來，A或B開頭，後面接五個數字，然後不管剩下的是什麼，全部都合併到後面那個正常的工具代碼，然後暖機前那個開機狀態利用關機超過24小時來過濾。

# 2017.07.27 (四)

切割機台STC資料發現，有一筆資料遺失使得我抓不到那個瞬間的工具代碼，我們有提出疑義，與家熙博士討論的結果正常來說不會有這種遺失狀況，目前採以手動排除，修改讀檔方式，先把所有資料全部開啟做分兩類(.txt)(.excel)做資料格式化，以往是手動填寫檔名，分辨不同機台資料。

# 2017.07.28 (五)

【公假：台北參加比賽(創新創業新秀選拔)】

# 2017.07.31(一)

期中報告，與長官們分享當前進度，開發工具與與資料處理方式，拋出目前發現的問題，當前DMG超前工單約2天，機台製作的零件有些根本沒有在工單排程裡，有插單的問題，不清楚熱機狀態該歸於誰較正確，目前是歸到後面零件的時間，STC機台所使用的程式不了解與這些程式該歸於誰，這攸關於往後建立時間模型正確性與準確性，讓生管能知道這個零件該給哪個機台做效率比較高，當天還有其他精彩的報告這裡不再多說明。

# 2017.08.01(二)

先前時間模型是以工具代碼與件號做值，做比對但是STC機台資料裡卻抓不到件號，改以只比對工具代碼，程式碼比對方式也稍作修改，原先是直接抓取供單資料一一去比對，改成建立資料結構後再做excel輸出，使得往後想使用這些資料時能夠方便取得而不是寫死，照成多撰寫一個方法使程式更加凌亂，可讀性降低。