**酷遊天股份有限公司**

**JTR**

**即訂即付API**

**System Analysis / System Design**

**documentation**

VERSION [1.0.0]

**[2018/08/20]**

| 程式代號 | JTR | 專案名稱 | JTR API Booking | SD/SA | 邱衿如,張峰榮 |
| --- | --- | --- | --- | --- | --- |
| 程式類別 | Web API | 使用技術 | ASP.NET Core | 審查者 | 胡良寬 |

|  |  |  |
| --- | --- | --- |
| 編製日期 | 修改內容 | 審核 |
| 2018/08/20 | Version 1.0完成 |  |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 需求確認簽名處 | | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目錄

[一． 開發目的： 3](#_Toc522884694)

[二． API流程圖： 3](#_Toc522884695)

[三． API串接資訊： 4](#_Toc522884696)

[五． 基本邏輯規則： 4](#_Toc522884697)

[六． 錯誤代碼匯整： 5](#_Toc522884698)

[七． KKDAY B2C API異常通知處理： 7](#_Toc522884699)

[八． KKDAY B2C API 取得回傳結果資料調整： 7](#_Toc522884700)

[九． KKDA Voucher 格式呈現： 8](#_Toc522884701)

[十． 程式開發架構： 9](#_Toc522884702)

[十一． B2B API Gateway分流機制設定： 9](#_Toc522884703)

[十二． B2S API JTR Booking 商編對應機制 ： 9](#_Toc522884704)

[十三． B2C API RQ與JTR Booking RQ mapping ： 9](#_Toc522884705)

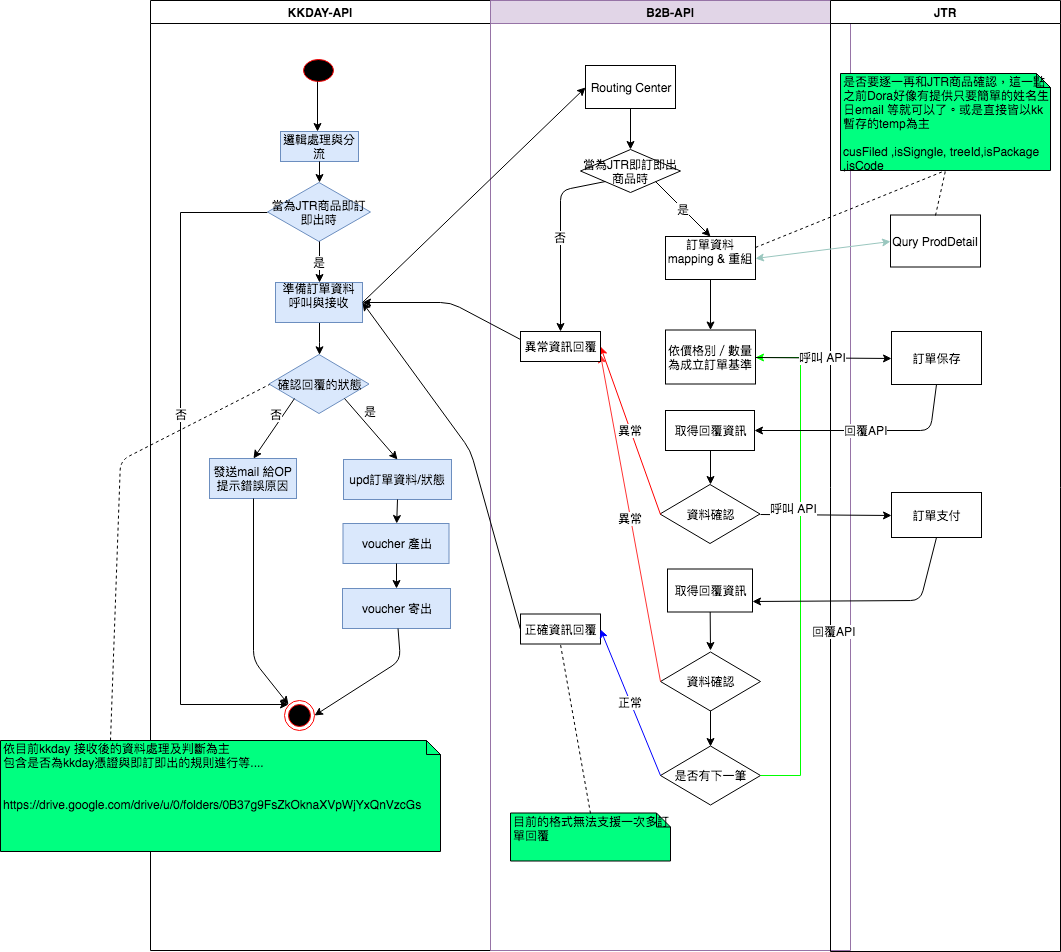
[十四． B2S API RS 定義 ： 10](#_Toc522884706)

[十五． 實際流程機制參考 ： 12](#_Toc522884707)

1. **開發目的：**
2. 即訂即出，節省客服與OP介入處理時間
3. 即訂即出，提高使用者體驗
4. **API流程圖：**

<https://drive.google.com/file/d/13sciBPhPebH0dhYnLB7KzJK3Ra6WSJ_r/view?usp=sharing>

1. B2C API 除了依 JTR 商品定義呼叫及取得B2B API資訊外，B2C API主要的流程依原設定不變



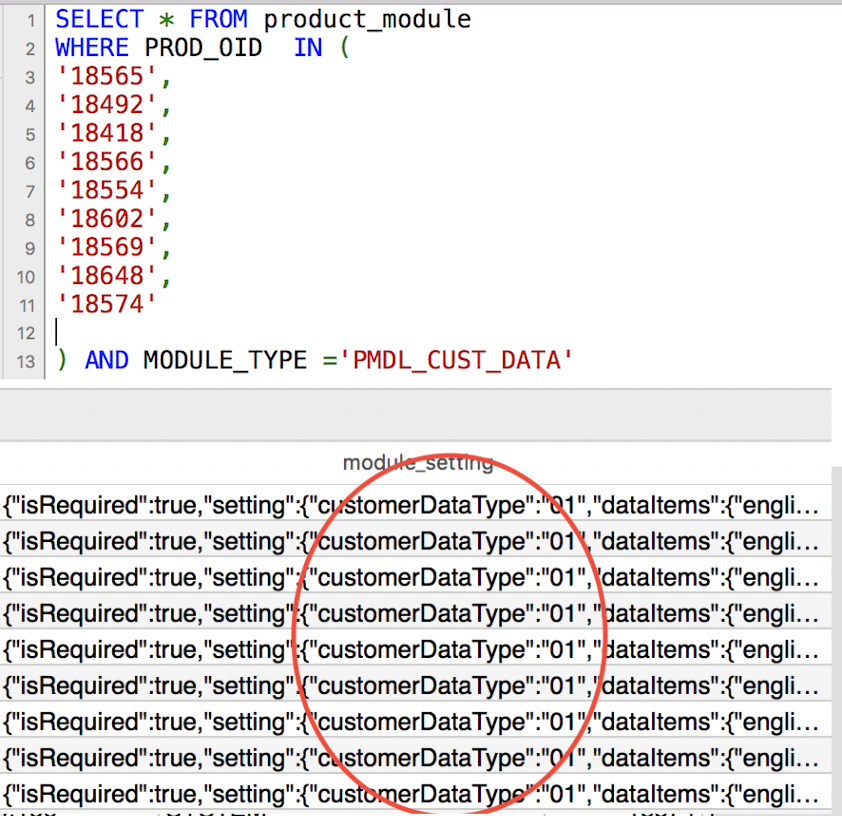
原描述流程連結：<https://drive.google.com/drive/u/0/folders/0B37g9FsZkOknaXVpWjYxQnVzcGs>

(2) B2B API 會重新定義 Gateway 位置，統一對接B2C API 接口，並由 Gateway進行未來各項API Booking 串接分流

1. **API串接資訊：**
2. B2C API 呼叫B2S API 所傳遞的訂單結構資訊文件：<https://drive.google.com/drive/u/0/folders/0B37g9FsZkOknaXVpWjYxQnVzcGs>

<https://docs.google.com/document/d/16tRTmqsEXpcmkPzqxcHtVNDPeienoU5p-Fm3pE8a93o/edit>

1. B2B API 呼叫JTR Booking的傳遞資訊文件：<https://drive.google.com/file/d/1nM7qVk4FAQnf9WmlKtY8pNtLhd6jVhNe/view?usp=sharing>
2. 這次主要完成的動作在於執行”訂單保存”與”交易支付”二個串接動作即可完成一次交易 ，（不額外再串接”異步出票通知” 原因為即訂即出，後續如交易成功但無12碼訂編時，OP人為介入處理）
3. JTR 需求文件：<https://drive.google.com/drive/folders/1k6K6Bgv-2GjkhZrC4yLUFTJphC-q4IP4>
4. **KKDAY & JTR雙方商編內容與對應：**
5. 此次串接商品由BD指定JTR 商品的mapping 位置如下：<https://docs.google.com/spreadsheets/d/1Bn4p76syL9XDYVc6bB6BNCVPfy6kxvLFANt09DGnyL4/edit#gid=891769266>
6. 雙方商品的對應關係為JTR 商編＝~~kkday商編 product\_oid +~~ 套餐編號pkg\_oid + 價格別
7. **基本邏輯規則：**
8. B2C API 呼叫B2B API timeout 時間為3分鐘，沒有回傳即為失敗
9. 一張KKDAY的訂單可能會對應到多張的JTR 訂編，回傳KKDAY 會重新定義json格式，~~若有多筆訂單時以<br/>區隔~~
10. 如果一張KKDAY有對應到多張JTR訂編時，在”訂單保存”時，皆直接將order\_mid +”-1 ” or +”-2” ，因為API無法多單指定單一訂編，如成人：18KK071987482-1 ，兒童18KK071987482-2
11. 透過”訂單保存”與”交易支付” ，會取得12碼的JTR 訂編，如果沒有取得即認定為失敗，後續由OP至JTR後台確認訂編 （不再串接異步出票通知）
12. 此次串接的商品，只要求客人填寫旅客代表人，故傳送時以旅客代表人為主



1. ”訂單保存” 接口Mail 統一以 [op-data@kkday.com](mailto:op-data@kkday.com) 為主
2. 之後如果要有新增的商品必須要重新檢視流程與程式
3. **錯誤代碼匯整：**

**JTR -> KKDAY B2B API**

|  |  |
| --- | --- |
| 代碼 | 說明 |
| 10001 | 产品已经无法获取当前日期的价格 |
| 10002 | 下单失败 |
| 10003 | 订单授信不足，不能支付 |
| 10004 | 支付订单失败 |
| 10005 | 订单已经不满足取消的条件，写入退改申请 |
| 10006 | 取消订单失败 |
| 10007 | 部分取消订单失败 |
| 10008 | 订单退改申请失败 |
| 10009 | 订单重发失败 |

**KKDAY B2B API -> KKDAY B2C API**

|  |  |
| --- | --- |
| 錯誤代碼 | 說明 |
| JTR-10001 | 產品己經無法獲取當前日期的價格 |
| JTR-10002 | 下單失敗  或  **1對多訂單類型，其中包含第1筆[KK訂編]支付成功[12碼訂編]，但第2筆成立訂單失敗，請OP至JTR後台協助確認**  **或**  **1對多訂單類型，第1筆下單即失敗** |
| JTR-10003 | 訂單授信不足，不能支付  或  **1對多訂單類型，其中包含第1筆[KK訂編] 支付成功[12碼訂編]，但第2筆[KK訂編]支付訂單失敗(**訂單授信不足，不能支付)**，請OP至JTR後台協助確認** |
| JTR-10004 | 支付訂單失敗  或  **1對多訂單類型，其中包含第1筆[KK訂編] 支付成功[12碼訂編]，但第2筆[KK訂編]支付訂單失敗，請OP至JTR後台協助確認**  **或**  **1對多訂單類型，第1筆[KK訂編]支付訂單失敗** |
| JTR-10005 | 訂單己經不滿足取消的條件，寫入退改申請 |
| JTR-10006 | 取消訂單失敗 |
| JTR-10007 | 部份取消訂單失敗 |
| JTR-10008 | 訂單退改申請失敗 |
| JTR-10009 | 訂單重發失敗 |
| JTR-10090 | Time out error |
| JTR-**10091** | **[KK訂編]支付成功，但未取得12碼供應商訂編，請OP至JTR後台協助確認**  **或**  **1對多訂單類型，其中包含第1筆[KK訂編]交易支付成功[12碼]，但第2筆[KK訂編]支付成功未取得12碼供應商訂編，請OP至JTR後台協助確認** |

1. **KKDAY B2C API異常通知處理：**

|  |  |  |
| --- | --- | --- |
| 錯誤代碼 | 說明 | 通知人 |
| 10001 | 產品己經無法獲取當前日期的價格 | OP |
| 10002 | 下單失敗 | OP,IT |
| 10003 | 訂單授信不足，不能支付 | OP |
| 10004 | 支付訂單失敗 | OP,IT |
| 10005 | 訂單己經不滿足取消的條件，寫入退改申請 | X (此次處理不包含取消) |
| 10006 | 取消訂單失敗 | X (此次處理不包含取消) |
| 10007 | 部份取消訂單失敗 | X (此次處理不包含取消) |
| 10008 | 訂單退改申請失敗 | X (此次處理不包含取消) |
| 10009 | 訂單重發失敗 | X (此次處理不包含取消) |
| 10090 | Timeout error | OP,IT |
| 10091 | **交易支付成功，但未取得12碼供應商訂編，請OP至JTR後台協助確認** | OP |

1. **KKDAY B2C API 取得回傳結果資料調整：**
2. 訂購成功：將供應商訂編寫入訂單主檔中
3. 並依流程決定狀態調整

|  |  |  |
| --- | --- | --- |
|  | Take Status | Process Status |
| 寄出憑證成功 | **已處理 3** | **已寄出Voucher 5** |
| 訂購失敗 | **新訂單已付款 1** | **客服未領取** |

1. **KKDA Voucher 格式呈現：**



1. 在右上角呈現 JTR 12碼訂編
2. 依JTR訂單數量呈現QR Code 數量
3. **程式開發架構：**
4. ASP.NET Core 2.1 WebAPI
5. Log4Net +ELK
6. AWS EC2
7. **B2B API Gateway分流機制設定：**
8. 依照B2C API中的order.packageOid進行分銷商booking分流比對

<item>  
    <kkday\_pkg\_oid>66658,63273,63275…</kkday\_pkg\_oid>  
    <sup>JTR</sup>  
</item>

1. **B2S API JTR Booking 商編對應機制 ：**
2. 依照B2C API中

Order. price1Qty大於0則kkday\_price\_type為price1

Order. Price2Qty大於0則kkday\_price\_type為price2

搭配訂購當下之package.oid找出對應之jtr\_prod\_no

<item>  
  <kkday\_prod\_pkg>63276</kkday\_prod\_pkg>  
    <kkday\_price\_type>price1</kkday\_price\_type>  
    <jtr\_prod\_no>11433215</jtr\_prod\_no>  
</item>  
  
<item>  
  <kkday\_prod\_pkg>63276</kkday\_prod\_pkg>  
    <kkday\_price\_type>price2</kkday\_price\_type>  
    <jtr\_prod\_no>11433864</jtr\_prod\_no>

</item>

1. **B2C API RQ與JTR Booking RQ mapping ：**

* B2C RQ與JTR order對應

|  |  |
| --- | --- |
| JTR order RQ | B2C RQ |
| info\_id | JTR商編 |
| cust\_id | 登入JTR ID |
| order\_source\_id | {order.orderMid} |
| num | {order.price1Qty}或{order.price2Qty} |
| travel\_date | {order.begLstGoDt} |
| link\_man | {order.orderCusList[0].cusLastname和order.orderCusList[0].cusFirstname } |
| link\_phone | 空字串 |
| link\_email | 指定op-data@kkday.com |

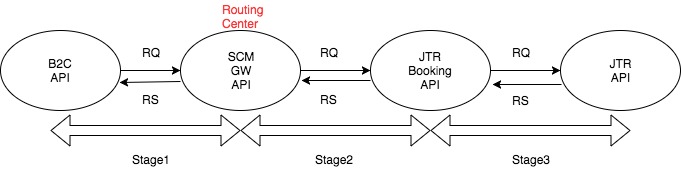
1. **B2S API RS 定義 ：**

* B2S RS參數說明

|  |  |  |  |
| --- | --- | --- | --- |
| 參數 | 類型 | 空值 | 說明 |
| metadata | Json |  |  |
| status | String | N | 訂購狀態代碼 ,如10004,100091 |
| description | String | N | 代碼說明 （註一） |
| data | Json | O | 若JTR第一筆訂單成立失敗則無回傳值 |
| isMuiltSupOrder | bool | N | 是否供應商為多筆訂單 |
| orderinfo | Array |  |  |
| priceType | String | N | 價格別  price1:大人（order.price1Qty>0）  price2:兒童（order.price2Qty>0） |
| priceQty | int | N | 價格別數量  1 , 2 |
| kkOrderNo | String | O | 此商品訂購的KKday訂編  第一筆為原本KK訂編-1  第二筆KK訂編-2  以此類推… |
| ticketNumber | string | O | JTR回傳12碼票號 |
| result | String | N | 目前有4種類型:  OK:此商品訂購付款取票成功  OrdErr:此商品訂購失敗  PayErr:此商品訂購成功付款失敗  NoTicket:此商品訂購付款成功取票失敗 |
| 參考內容: | | | |
| {  "metadata": {  "status": "10004",  "description": "1對多訂單類型，其中包含第1筆(KK訂編)支付成功)，但第2筆(KK訂編)交易訂單支付失敗，請OP至JTR後台協助確認"  },  "data": {  "isMuiltSupOrder" :true,  "orderinfo": [  {  "priceType": "price1",  "priceQty": 2,  "kkuniOrderNo": "18KK071987482-1",  "ticketNumber": "1234567891234",  "result" :"OK"  },  {  "priceType": "price2",  "priceQty": 1,  "kkuniOrderNo": "",  "ticketNumber": "",  "result" :"OrdErr"  }  ]  }  } | | | |

（註一）：B2C API 可以以此訊息內容做為回覆OP的內部訊息

1. **實際流程機制參考 ：**



stage1🡺依照B2C API RQ之json格式用POST方法呼叫SCM Gateway API

RQ：請參考『KKDAY API查詢 Ticketing Mapping 表並做訂購』

[https://docs.google.com/document/d/16tRTmqsEXpcmkPzqxcHtVNDPeienoU5p-Fm3pE8a93o/edit - heading=h.oaawaa0u89c](https://docs.google.com/document/d/16tRTmqsEXpcmkPzqxcHtVNDPeienoU5p-Fm3pE8a93o/edit#heading=h.oaawaa0u89c)

RS：同stage2 RS

stage2🡺透過分流機制，呼叫B2S JTR Booking API

RQ：

依照B2C API 的Request內容再加上sup\_id及sup\_key屬性

{

"sup\_id":"801790",

"sup\_key":"616CDBF19C4740E3B7230F41410C28FB"

}

以上是JTR測試ID及KEY

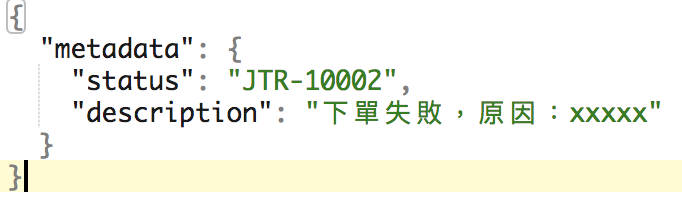
RS：內容如下：

訂購成功回覆:



訂購失敗:

訂單成立失敗回覆：



訂購成功但未取得12碼供應商訂編



一對多訂購失敗（並非JTR全部商品訂購成功）

其中一筆訂購失敗/付款失敗



其中一筆訂單成功付款成功但第2筆訂單成功但取票失敗



stage3🡺

傳入的RQ ：同stage2 RS

依照pogoed + 價格別(Price1Qty,Price2Qty)找出JTR Prod No以及各個價格別之數量

"order info": [

{

"price type": "price1",

"jtr\_prod\_no": "10457115",

"qty": 2

},

{

"price\_type": "price2",

"jtr\_prod\_no": "10457114",

"qty": 1

}

]

依照order\_info筆數跑迴圈執行**訂單保存接**口及**訂單支付接口，**若中途有失敗訊息則跳出迴圈

＊目前商品結構單純暫不需要執行JTR PROD QUERY找尋custField內容之步驟

orderRQ：

{

"info\_id":"10457115",

"cust\_id":"801790",

"order\_source\_id": "17KK081858222",🡺unique

"num": "2",

"travel\_date":"2017-08-31",

"link\_man":"chen pin",

"link\_phone":"886-29951234",🡺空值

"link\_email":op-data@kkday.com 🡺指定給OP

}

orderRS：

{

"result":

{

"status":1,

"msg":"订单已经提交",

"order\_id":"66692410",

"error\_state":"10000",

"error\_msg":"单已经提交",

"order\_money":4,

"mem\_order\_money":6,

"order\_state":3

}

}

payRQ：

{

"order\_id":"66692410"

}

payRS：

{

"result": {

"orderId": "66861810",

"status": "1",

"msg": "支付成功",

"code": "668618106800",

"error\_msg": "支付成功",

"error\_state": "10000"

}

}

傳出的RS：

同stage1 RS