|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1 Version | 13-04-2021 | Santosh Reddy | * As per Item-3 (20981) Country requirement, We increased the numFiles column from 1 to 2 for the Two E1 and US Regions. * The Output files are generated in their corresponding target path for both E1 and US Regions.   Check PageNo:6 (section 1.1.2.1) |

## General Specifications

This section covers the design / customization made based on the specifications given in the FS. The table below provides a guide to finding the Talend job that implements the described logic. For jobs where only the name of the job is given, the design is simple enough that the field level detail given in the functional specification is the full detail required to fully understand the design of the transformation and no additional detail is given here. For the others that require further explanation, a further subsection has been defined below.

|  |  |  |
| --- | --- | --- |
| **FS Name** | **Talend Job** | **Detail Subsection** |
| FS-TLD-Control Tables (Site.tab, Buyer Code.tab, Planner Code.tab, material Type.tab, MRP Group.tab) | Site – N/A  TR\_BuyerCode  TR\_PlannerCode  TR\_MaterialType  TR\_MRPGroup | n/a |
| FS-TLD-CalendarDate.Tab | TR\_Calendar | Refer to the sections below. |
| FS-TLD-Part.tab | PreTR\_Part  PreTR\_Part\_Consign  PreTR\_Part\_GlobalMaterial  TR\_Part | Refer to the sections below. |
| FS-TLD-Supplier.tab | TR\_Supplier | n/a |
| FS-TLD-Part Source\_Transfer/buy.tab | TR\_PS\_Tr\_Buy  TR\_PS\_Tr\_Buy\_Quota  TR\_PS\_Tr\_Buy\_SrcLst\_IM  TR\_PS\_Tr\_Buy\_SrcLst\_SZ  TR\_PS\_Tr\_Buy\_Subco  TR\_PS\_Tr\_Buy\_Subco\_SPK | Refer to the sections below. |
| FS-TLD-PartSource\_Make.tab | TR\_PS\_Make | Refer to the sections below. |
| FS-TLD-Bill Of Material.tab | TR\_BOM | Refer to the sections below. |
| FS-TLD-Customer.tab | TR\_Customer | n/a |
| FS-TLD-SalesOrderType.tab | TR\_SAPOrderType | n/a |
| FS-TLD-ProcessOrdertype.tab | TR\_SAPOrderType | n/a |
| FS-TLD-PurcharseOrdertype.tab | TR\_SAPOrderType | n/a |
| FS-TLD-Location.tab | TR\_Location | n/a |
| FS-TLD-UnitofMeasure.tab | TR\_UnitOfMeasure | n/a |
| FS-TLD-PartUOMConversion.tab | TR\_PartUOMConversion | n/a |

### Talend BI File Processing

The Talend transformation process works on a set of files that have each been constructed from the individual files sent from each region by BI. There are two distinct steps to maintaining those files

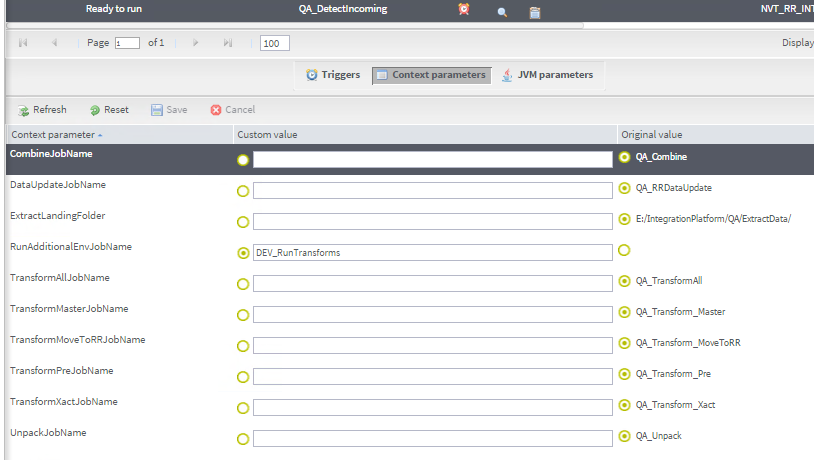
1. Update the working set with latest BI file delivery
2. Combine the individual region files to one overall file

The overall Talend transform process is controlled by the UTIL\_DetectIncoming job. It looks for incoming loads of data from BI, calls the Unpack Job to process the new files into the working set, and then calls the appropriate jobs to run the transforms and load the data to RR. The additional jobs used are:

* EX\_Unpack – Processes incoming files from BI into the current working set. Functionality of this Job is described below.
* UTIL\_CombineFiles – Combines the files in the working set into composite files used by the Transform jobs. Functionality of this Job is described below.
* Transform\_Pre – Preprocessing to create intermediate files used by downstream jobs. This job contains support functionality for both Master and Transactional jobs.
* Transform\_Master – Produces master data files (Part, PartSource, etc.). The sub jobs called by this Job are covered by *DS-TLD-Talend Transformations for group 1* and the associated FS documents.
* Transform\_Xact – Produces transactional data files (SR\_\*, IndDmd\_\*, etc.). The sub jobs called by this Job are covered by *DS-TLD-Talend Transformations for group 2* and the associated FS documents.
* UTIL\_MoveToRR – Copies all generated files from Talend IntegrationPlatform to RR Extract Directory. Functionality of this Job is described below.
* UTIL\_RRDataUpdate – Starts the RR Data update process. Functionality of this Job is described below.

The DetectIncoming Job is designed for deployment into the TAC, so each of the above jobs should be deployed individually into TAC and given a name specific to the environment (DEV, QA, PROD) and the specific name filled into the parameters. These jobs are called indirectly using the UTIL\_Run\_TAC\_Task job. **That job should return an appropriate error in the child job fails.**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter Name | DEV | QA | PROD |
| UnpackJobName |  | QA\_Unpack | PROD\_Unpack |
| CombineJobName |  | QA\_Combine | PROD\_Combine |
| TransformPreJobName | DEV\_Transform\_Pre | QA\_Transform\_Pre | PROD\_Transform\_Pre |
| TransformMasterJobName | DEV\_Transform\_Pre | QA\_Transform\_Master | PROD\_Transform\_Xact |
| TransformXactJobName | DEV\_Transform\_Xact | QA\_Transform\_Xact | PROD\_Transform\_Xact |
| TransformMoveToRRJobName | DEV\_Transform\_MoveToRR | QA\_Transform\_MoveToRR | PROD\_Transform\_MoveToRR |
| DataUpdateJobName | DEV\_RRDataUpdate | QA\_RRDataUpdate | PROD\_RRDataUpdate |
| RunAdditionalEnvJobName |  | DEV\_RunTransforms |  |

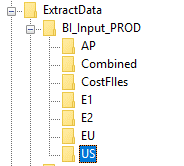


Each time the DetectIncoming job runs, it first checks for a new directory of files delivered from the sFTP landing zone. If it finds a directory to be processed, it calls Unpack, and then places an unpack.done file in the extract root directory. If no directory is found to be processed, it looks for unpack.done. If that is found, then it calls the transform processes (Combine, Pre, Master, Xact, MoveToRR, RRDataUpdate). If no new directory and no unpack.done is present the job exits. If any of the child jobs fail, an email is sent to a configurable address or addresses.

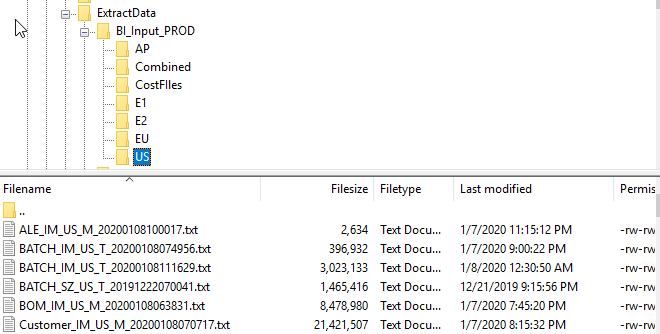
To provide operation flexibility, DetectIncoming calls one additional TAC job after the transform process jobs complete. The concrete example of this is running the DEV environment transforms tied to the same data used in QA. This is shown in the above screenshot with DEV\_RunTransforms configured in the RunAdditionalEnvJobName parameter.

**Working Set Structure**

The working set is stored on disk in the Kinaxis IntegrationPlatform in a series of subdirectories. The structure is as follows:



The individual files delivered from BI are stored within the subdirectory that correspond to the region of the files themselves. The following example shows a few of examples of the US files sitting in the US subdirectory:



### Processing Jobs

#### Talend Job: EX\_Unpack

The unpack job serves 2 functions:

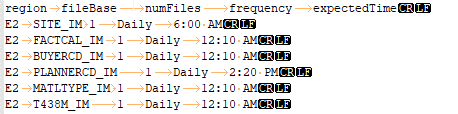
1. Unzip files received from BI .zip files
2. Correctly place the received files into the working set

The overall flow of the job looks like this:



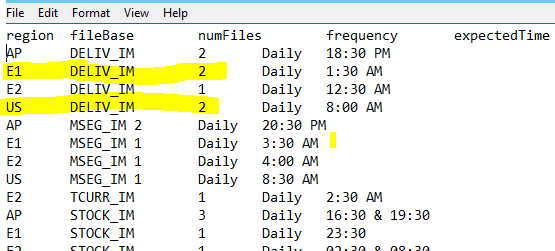
Notes:

1. Each incoming file has structure <Base>\_<Region>\_<Type>\_<DateTime Stamp>.txt (eg: BATCH\_IM\_US\_T\_202001008074956.txt). Each set of files received is for only one region, and the delivery is expected to contain all data for that region which might consist of multiple files.
2. If the process finds any files with a given base in the incoming set, all files with that base will be replaced within the regional directory.
3. The file ../IntegrationPlatform/<Env>/PersistedData/ExpectedFiles.txt defines the number of expected files of each base type that are allowed. If the process finds more than the expected number of files, the files are ordered by their timestamps and only the expected number are selected starting from the most recent backwards. This is to prevent duplicate records when the BI process delivers multiple sets of files in one payload. The structure of the expected files file shown here -

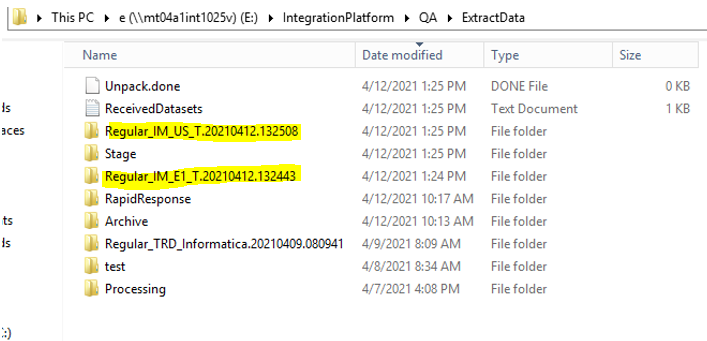


As per the requirement (20981), We have increased the “**numFiles**” column from 1 to 2 for E1 and US Regions in “ExpectedFiles.txt” as shown in below screenshot.

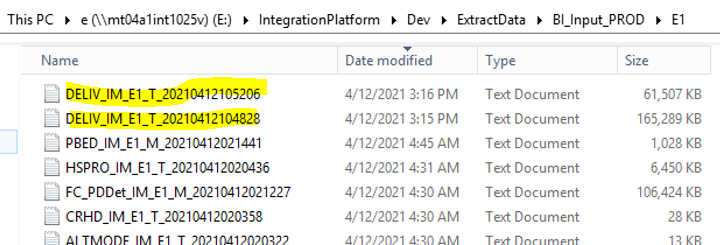
E:\IntegrationPlatform\PersistedData\ExpectedFiles.txt

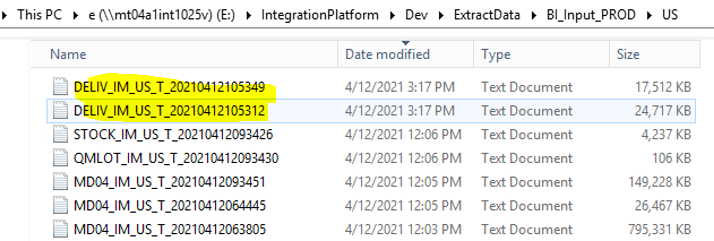


These are source files as mentioned below screenshot

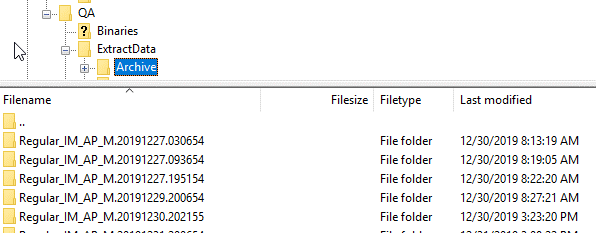


These are Target files, Output files are generated in their corresponding target path for E1 and US Regions as shown below.





4.The incoming file sets are archived to /IntegrationPlatform/<Env>/ExtractData/Archive/



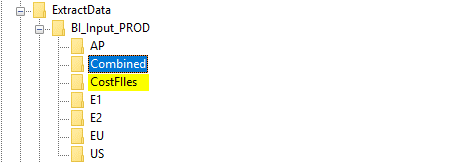
|  |  |
| --- | --- |
| Parameter | Description |
| BasePath | Extract Data root path where job looks for incoming file sets |
| DestinationBasePath | Root folder of the working set where incoming files are placed |
| PersistedDataPath | Location of expectedFiles.txt file |

#### Talend Job: UTIL\_CombineFiles

The combine job takes all files of common base from all regional subdirectories of the working set and combines them into a single file of the form <Base>.txt. Any Cost files are moved to the Cost files directory. Flow of job shown here:



Placement of Combined & Cost files:



#### **Talend Job: UTIL\_MoveToRR**

This job runs after all other Transformation jobs are complete. It moves all generated files from the Talend Processing area to the RR Extract folder for RR processing.



Notes:

1. The Talend job TR\_DataUpdateFileSet is called to complete this. This job lists all the files in the current working set and creates data file DataUpdateFileSet.tab to load in RR.

|  |  |
| --- | --- |
| Parameter | Description |
| RapidResponseExtractFolder | Network directory destination location for files. |
| TransformRootFolder | Source directory for data files |

#### Talend Job: UTIL\_RRDataUpdate

This is a one step Talend job that connects to RR and triggers the Data Update process to start.

|  |  |
| --- | --- |
| Parameter | Description |
| Endpoint | URL for the instance of RR where the data update should be started |
| IntegrationScenario | Not used |
| KeepExtract | Passthough to Data Update process. Tells RR to keep the data files after completion of the update or not. |
| OutputDirectory | Not Used for this job |
| RRApplicationServer | Not Used for this Job |
| RRPassword | Password of the RR Web API user |
| RRUsername | User name of the RR Web API user |
| Scenario | Scenario where the Data Update process should look for Data Sources and Mapping information |
| TargetScenario | Target scenario for the incoming data |

#### Talend Job: UTIL\_Run\_TAC\_Task

This Talend job connects to Talend TAC and triggers a job configured there to run. The result should be returned.

|  |  |
| --- | --- |
| Parameter | Description |
| metaServlet\_taskName | Name of the TAC task to be started |
| metaServlet\_tacUrl | URL of teh TAC server to connect to |
| metaServlet\_mode | Mode of the task start (synchronous or asynchronous) |
| metaServlet\_user | User account to use for connecting to TAC |
| metaServlet\_password | Password of the user account |

### Integration Platform Maintenance

The incoming data files are archived as described in the EX\_Unpack job description. Over time, left un managed, these archived files will fill up the disk space available in the Integration Platform. To maintain a reasonable disk availability, an archive job is created and scheduled.

#### Talend Job: UTIL\_FileCleanup

This job looks at a directory and removes subdirectories older that X days old. The job flow looks like this:



|  |  |
| --- | --- |
| Parameter | Description |
| ArchiveDirectory | Root directory containing archived directories to be purged |
| KeepDays | Number of days old an archive directory can be and still be in the Archive directory. All directories older will be removed. |

## Business Use Case / Module Settings

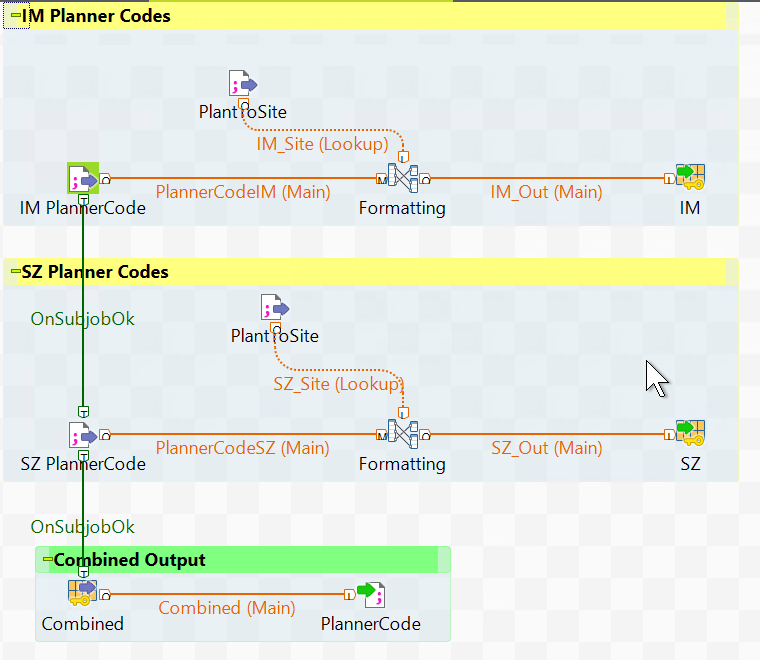
The business logic for these interfaces are given in the individual FS documents. Where appropriate, the functionality is further detailed in section below.

## Interfaces

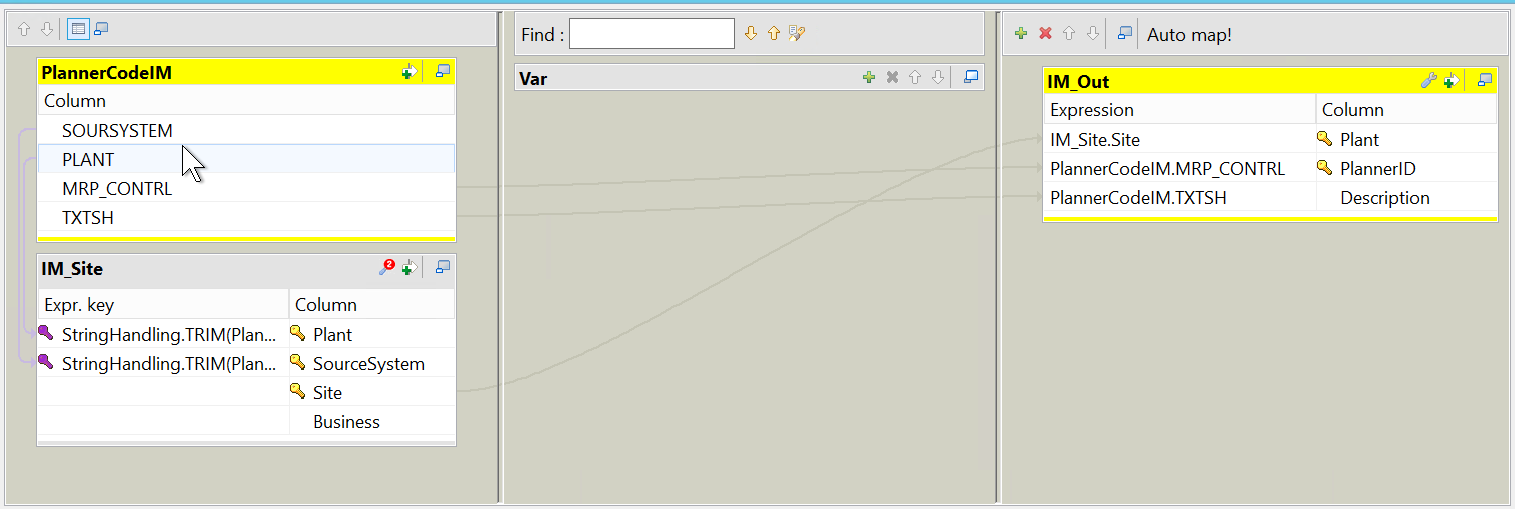
This section covers the interface design / customization made based on the specifications given in the FS

### General Talend Flow

Since most specifications/BI input is separated for IM vs SZ, the general jobs will have the format of separate logic threads for the individual business units. Shown here for the typical case using the example of planner codes.



Further, within the Talend tMap components (labelled above as Formatting), the common cases implement the details of the functional specification.



The PlantSite file shown above as IM\_Site is used throughout all Talend jobs to correctly convert the input PLANT and SOURCE SYSTEM to the corresponding RR Site.

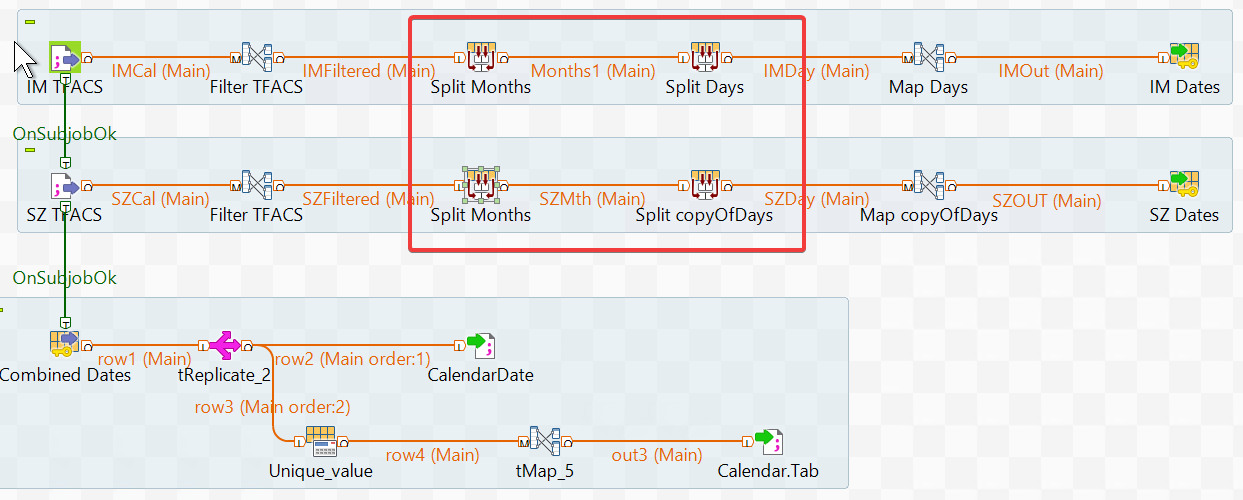
Context variables are used for managing the different parameters that drive the configuration of the flow of data.

|  |  |
| --- | --- |
| **Name** | **Value** |
| DebugRootFolder | E:/IntegrationPlatform/Dev/IntermediateData/Debug/ |
| ExtractRootFolder | E:/IntegrationPlatform/Dev/ExtractData/ |
| Input\_FieldSeparator | "\t" |
| IntermediateFolder | E:/IntegrationPlatform/Dev/IntermediateData/ |
| LineSeparator | "\r\n" |
| RapidResponseExtractFolder | \\\\ch2p2svm1rr0\\rr\_nvtd01\_dev\_dis\\RapidResponse\\Extract\\Novartis\ |
| RR\_FieldSeparator | "\t" |
| TransformControlSet | Default |
| TransformRootFolder | E:/IntegrationPlatform/Dev/PersistedData/ |

### CalendarDate

Implementation details for FS-TLD-CalendarDate.Tab

The conversion of the calendar information from the SAP provided from SAP to the format required by RR involves a small amount of logic. The SAP calendar comes in a very compact format that consists of one row with 12 monthly fields. The monthly fields contain 31 sub-fields that represent the calendar as being open or closed for that day of the month. The Talend logic converts each of those month field/day bitmap combinations into a record (if open) for consumption by RR. Talend has specialized components for changing fields within a row into multiple output rows. Those are used here as shown below.



In addition to the incoming Plant calendars, the Calendar job is also responsible for updating the dates of standard RR calendars: Month, Quarter, Workday, Year, Week. These are based on the context parameter values PreviousCalendarYears and MaxCalendarYears

Month: Value - First day of every Month; Display Value – mmm yy (ex. Apr 20)

Week: Value – First Monday of every week; Display Value – n/a

Year: Value – First day of every year; Display Value – yyyy (ex. 2020)

Workday: Value – Every day excluding Sat, Sun; Display Value – YYYY-MM-DD

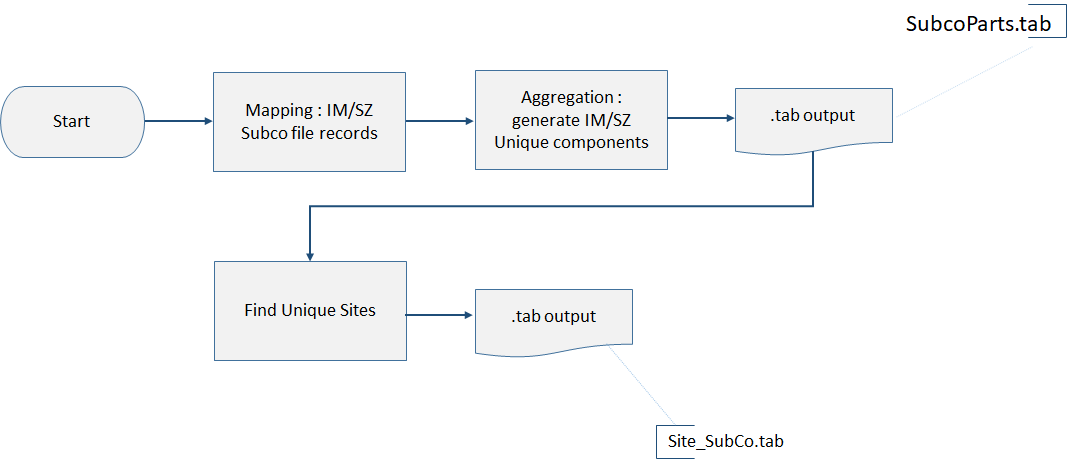
Quarter: Value – First day of each calendar quarter; Display Value – Qx YY (Ex. Q1 20)

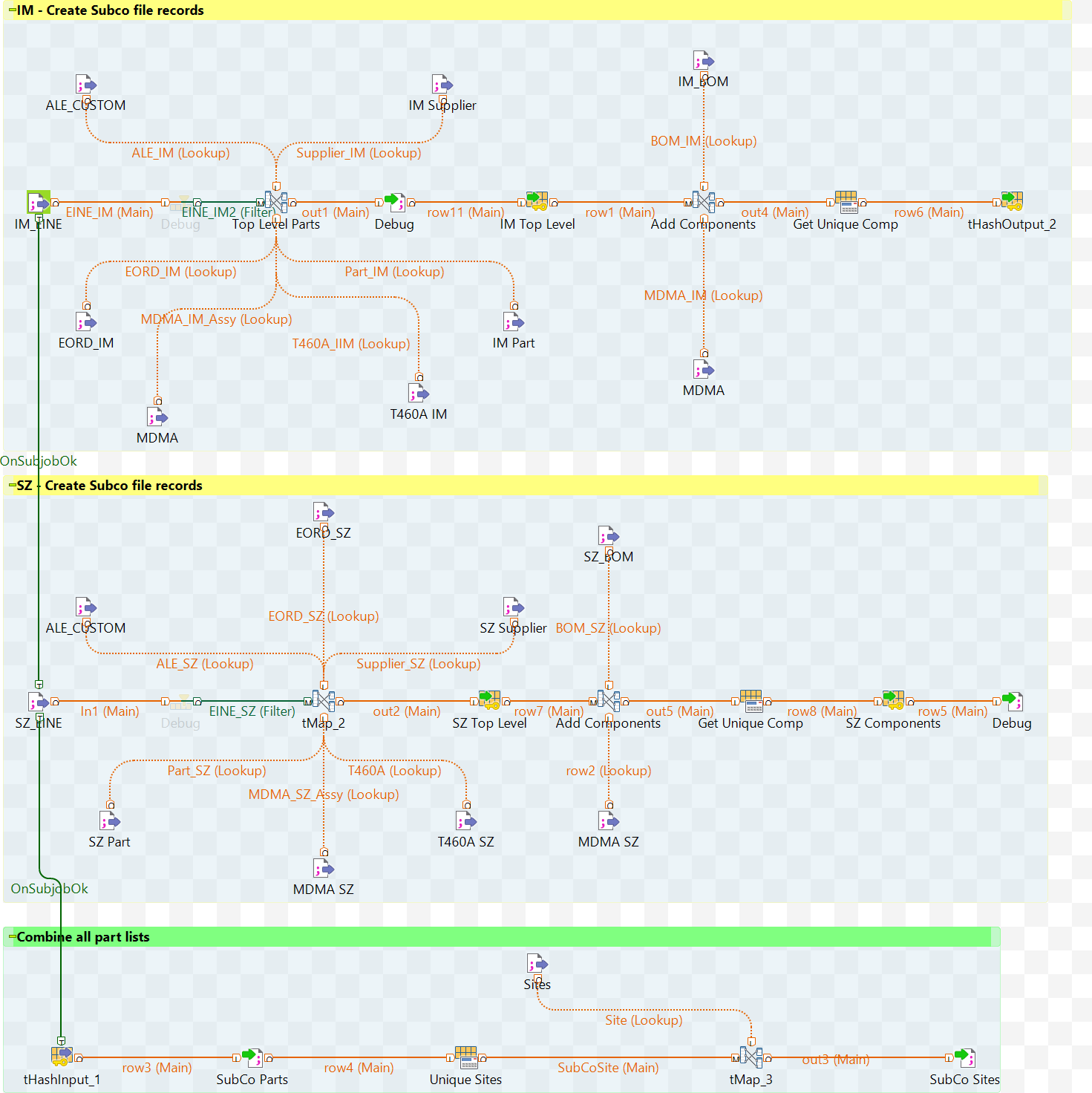
### Part

Implementation details for FS-TLD-Part.tab

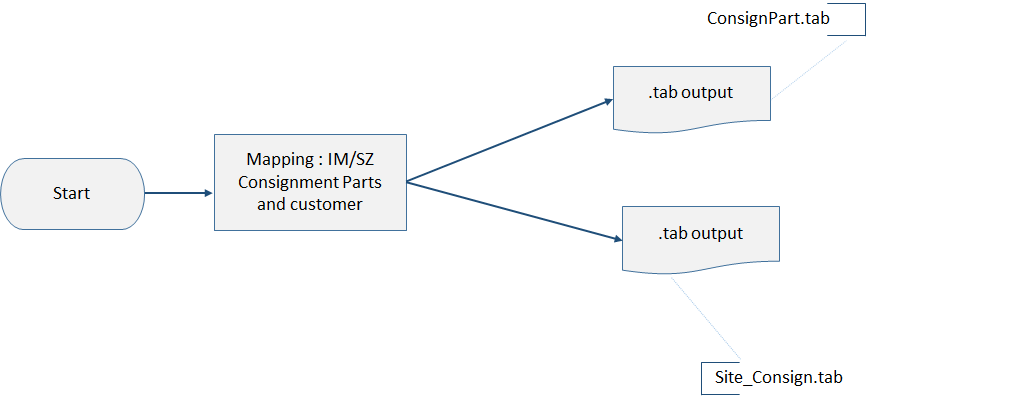
The TR\_Part job encompasses these basic steps. The field detail of the mappings is described in the FS. Apart from the main job, there are multiple other jobs for Part, each handling different modules viz. Subcontracting, Part List, Consignments. These supporting jobs read the data provided from SAP and convert them into a format compatible with RR. The output .tab files that these jobs generate are then used in the main job to extract the required information.

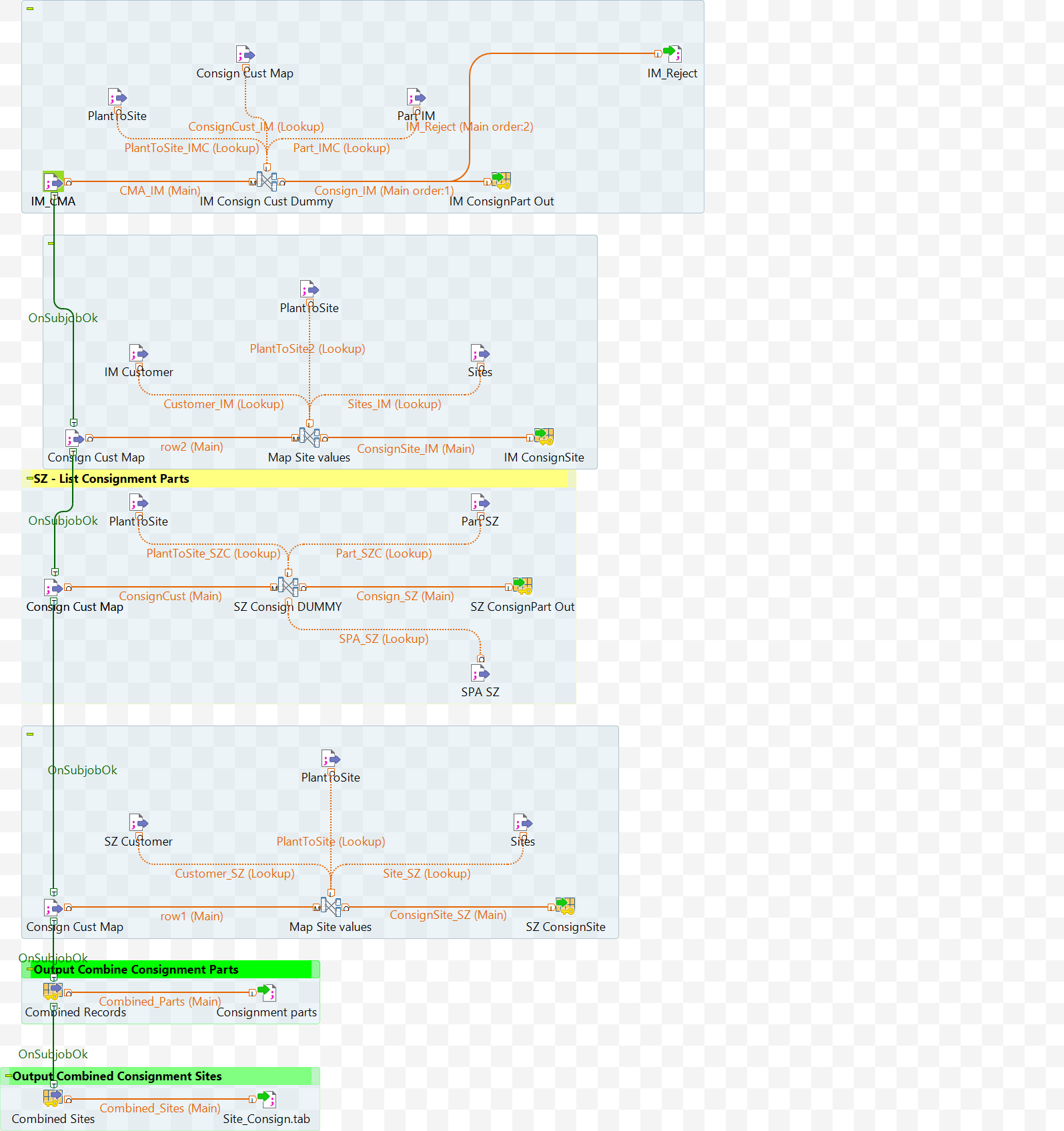
PreTR\_Part :

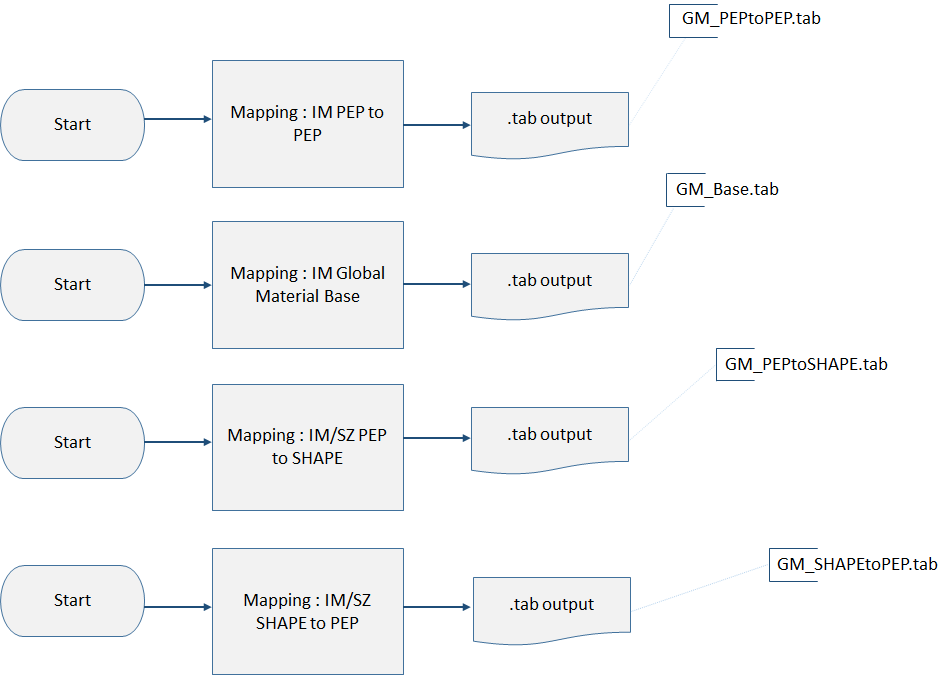


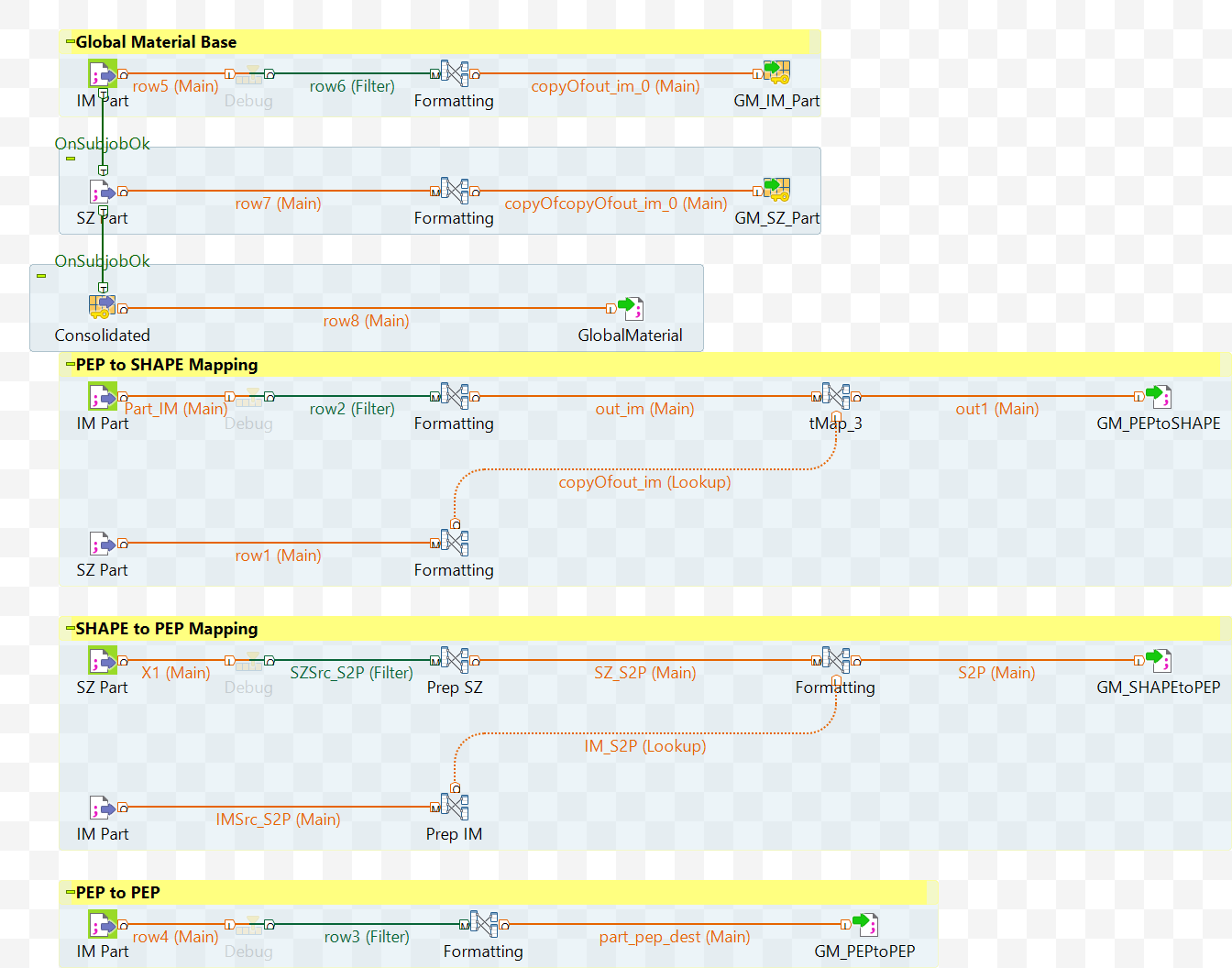


PreTr\_Part\_Consign:

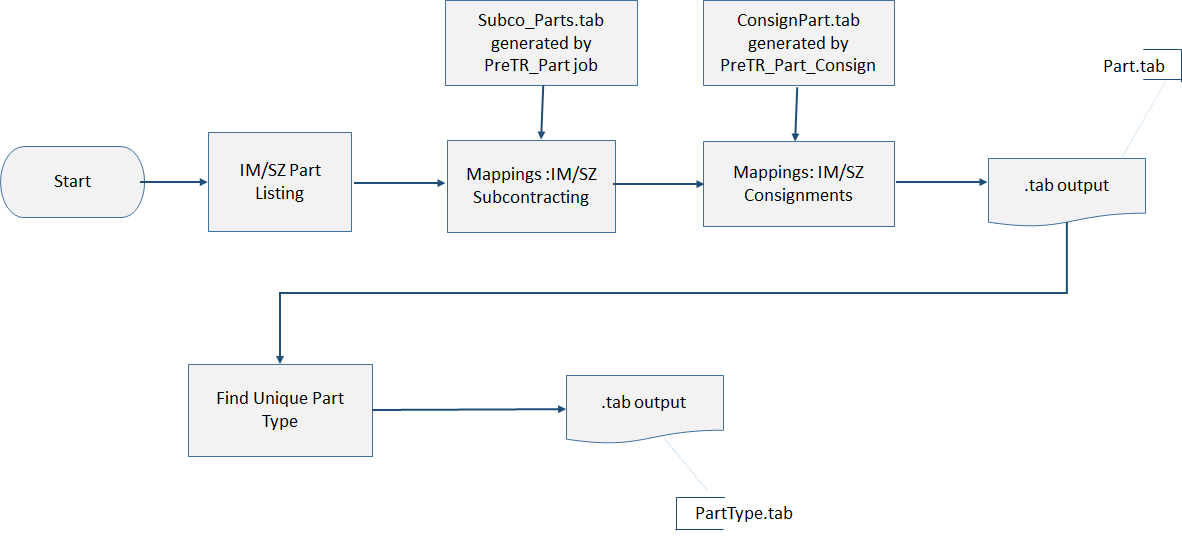


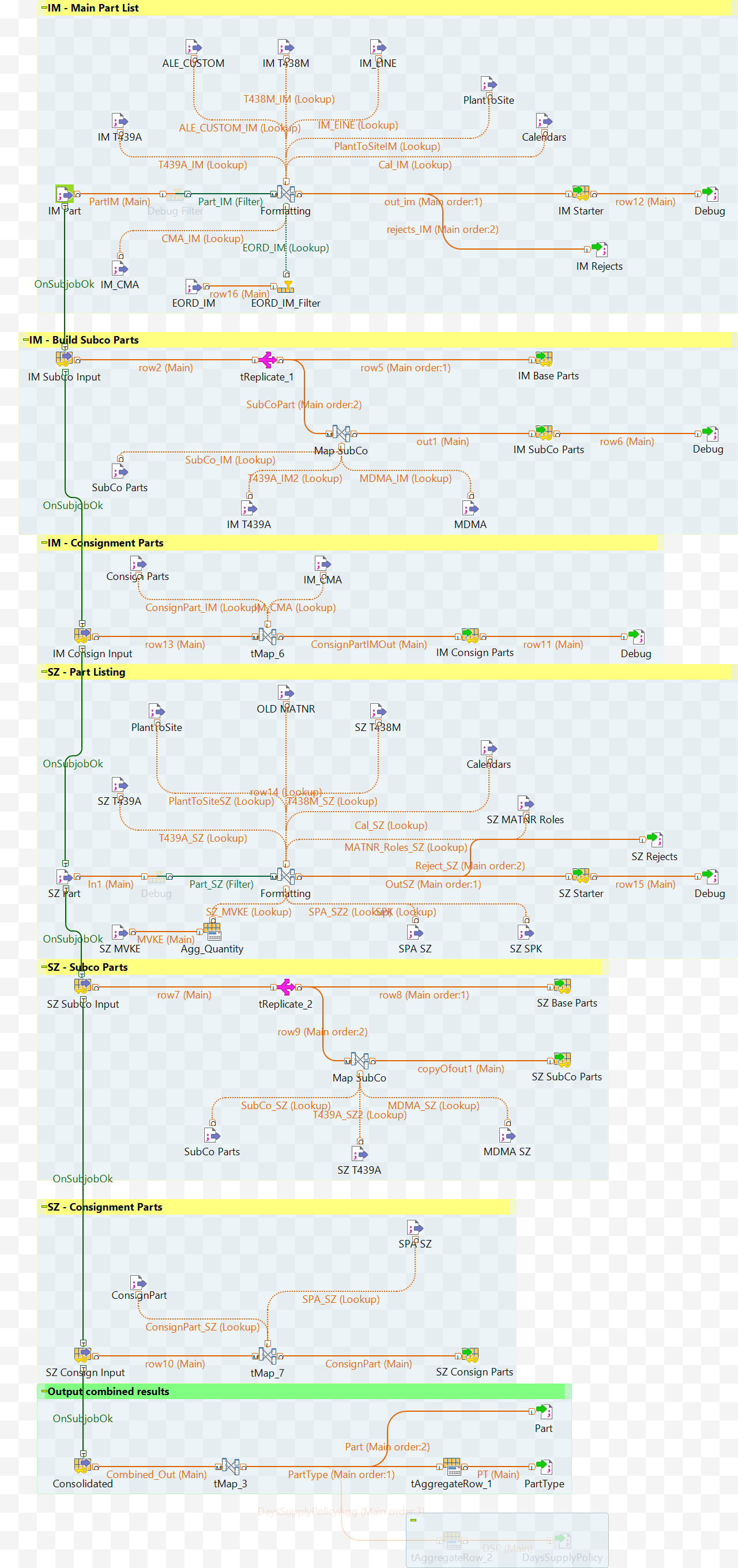


PreTR\_Part\_GlobalMaterial:



TR\_Part :

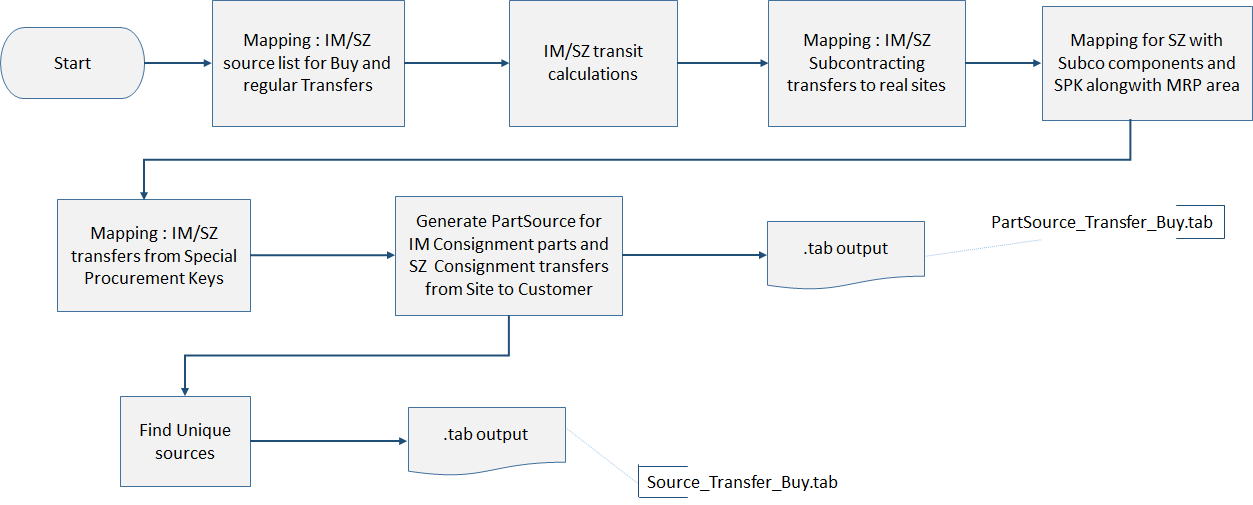


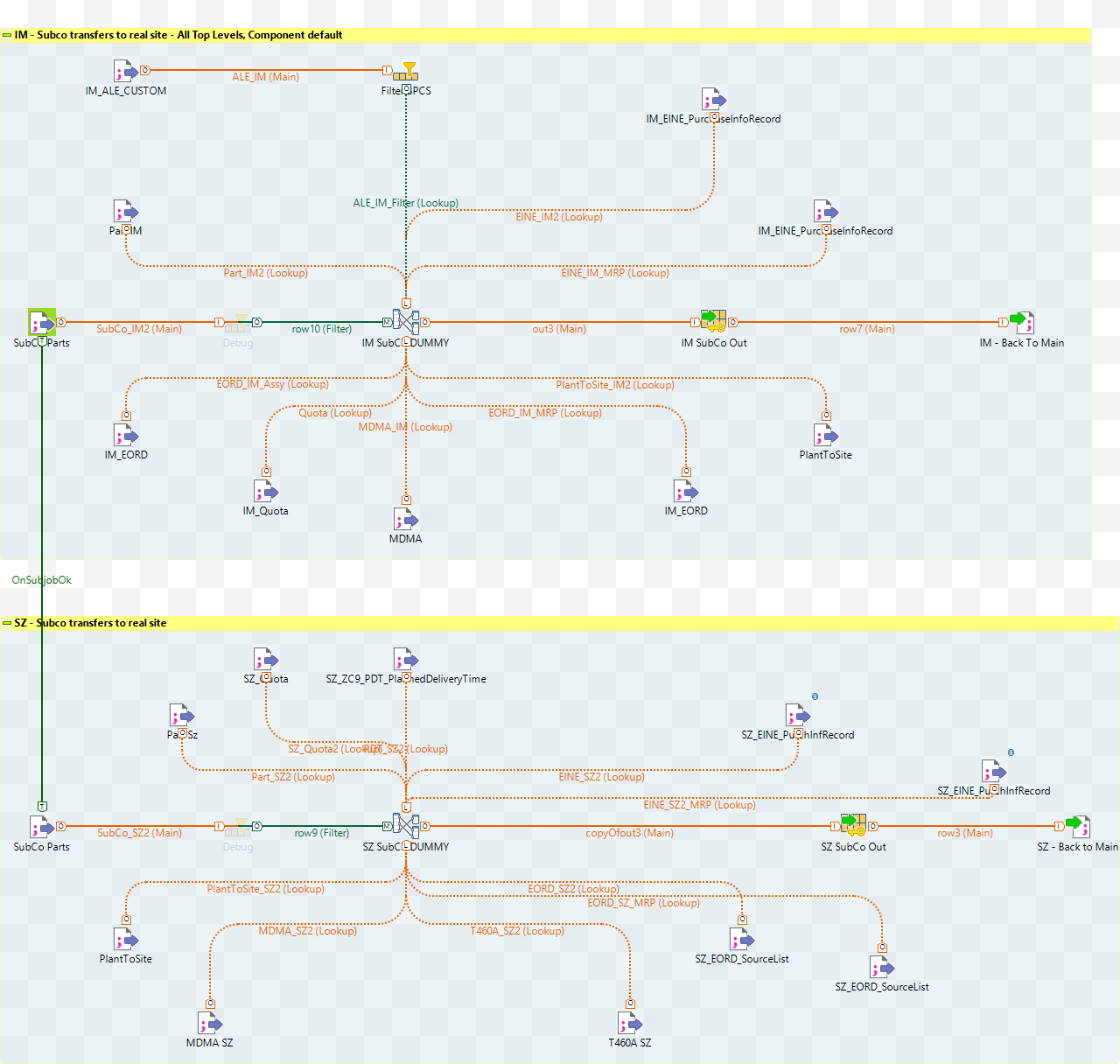


### PartSource – Transfer

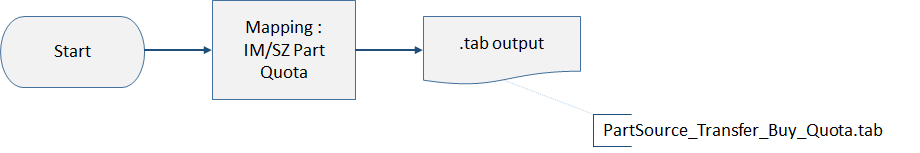
Implementation details for FS-TLD-Part Source\_Transfer\_buy.tab

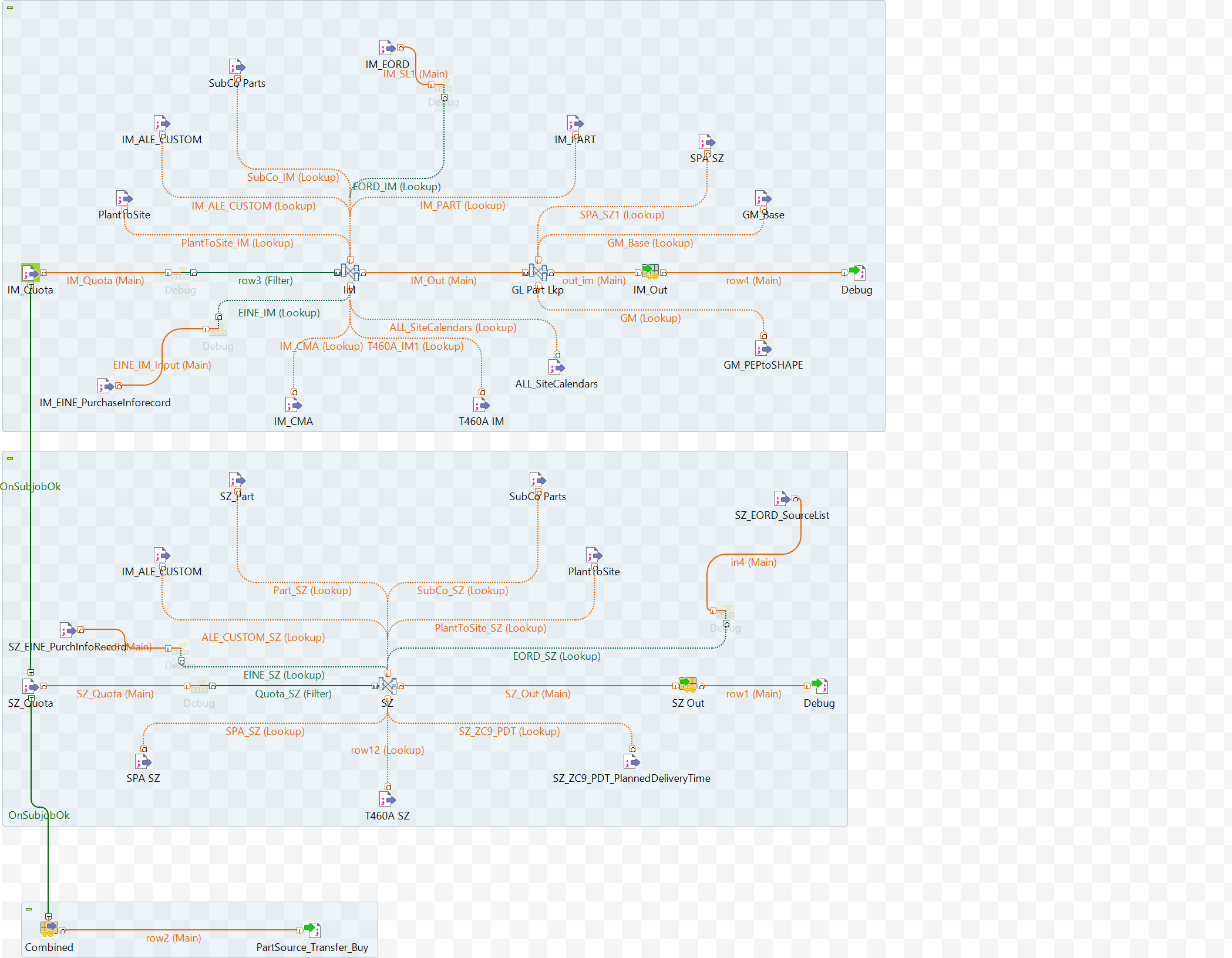
The TR\_PS\_Tr\_Buy job encompasses these basic steps. The field detail of the mappings is described in the FS. Apart from the main job, there are multiple other jobs for PartSource\_Transfer\_Buy, each handling different modules viz. Subcontracting, Source List, Quota, Subcontracting with Special Procurement Key. These supporting jobs read the data provided from SAP and convert them into a format compatible with RR. The output .tab files that these jobs generate are then used in the main job to extract the required information.





TR\_PS\_Tr\_Buy\_Quota:





### PartSource - Make

Implementation details for FS-TLD-PartSource\_Make.tab

The TR\_PS\_Make job encompasses these basic steps. The field detail of the mappings is described in the FS.



|  |  |
| --- | --- |
| Reference | Note |
| 1 | A fallback priority described in the FS is based on an ordered sort of production versions. Since this logic requires information outside about an individual record’s relationship to other records, it needs a pre-processing step. The information is then used within the mapping steps |
| 2 | Some Parts, whose manufacture is subcontracted, do not have production versions within SAP. For these, additional Part Sources must be generated. The basis for this generation is the listing of sub-contracted parts from the SubCoParts.tab file .Only the parts not covered by a valid production version need to be generated. The mapping uses only the fields that can be mapped from the xx\_Part.txt inputs from the FS. |
| 3 | A Rapid Response Part Source refers to a sub object (Source) that contains specific information on the site->site link. This information is fed to RR in a separate file, but since the data source from SAP is the same, we combine the output with Part Sources for simplification. |

### 

### BillOfMaterial

Implementation details for FS-TLD-BillOfMaterial.tab

The BillOfMaterial job encompasses these basic steps. The field detail of the mappings is described in the FS. The TR\_BOM job reads data provided from SAP and transforms it into a format compatible with RR.

