**Service Billing Main Process**

* **Overview - Pull work orders from Thirsty and update in S4 Hana.**
* **KEY SCENARIOS** 
  + Part A – Regular : Pick Work Orders which are ‘Approved for AP’ (see APL\_Query)
  + Part B – Accruals : Pick Monthly for Work Orders which are ‘Not’ Approved for AP.
  + Backfeeds to Thirsty – Discussed in Separate Document
  + Parts Sales – Discussed in Separate Document
* **WORKFLOW GROUPS AND SCHEDULING INFO** 
  + **TCCC\_ServiceBilling.WFG\_ServiceBilling**

Enabled in Prod to run Every 4 hrs

* + - TCCC\_ServiceBilling.WFL\_LoadGroupingID.workflow\_1
    - TCCC\_ServiceBilling.WFL\_ThirstyCollect.Billing
    - TCCC\_ServiceBilling.WFL\_BP\_LookUp.FWD1
    - TCCC\_ServiceBilling.WFL\_PaySP.FWD1
    - TCCC\_ServiceBilling.WFL\_CustomerBilling.FWD1
    - TCCC\_ServiceBilling.WFL\_WarrantyClaims.Process
    - TCCC\_ServiceBilling.WFL\_InvoiceStatus\_SplitFiles.workflow\_1
    - TCCC\_ServiceBilling.WFL\_InvoiceStatus.Process
  + **TCCC\_ServiceBilling.WFG\_ServiceBilling\_RT**
    - All TR WFL related to above Batch
  + **TCCC\_ServiceBilling.WFG\_ServiceBilling\_Accurals**
    - Scheduled by AutoSys
    - TCCC\_ServiceBilling.WFL\_ThirstyCollect.Accurals
    - Only Customer Billing
    - No Warranty , No PaySP, No Invoice Status back to Thirsty – since invoice not paid OR approved for ap
  + TCCC\_ServiceBilling.WFG\_SB\_repro(1)
    - Not Started
    - TCCC\_ServiceBilling.WFL\_InvoiceStatus.Reprocess(4)

**File Paths and Debug File Information:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| AP Payment | This 200 code means success  Response:  Field Values for: RESTClient.RESTResponse [com.digitalroute.mz.rest.types.RESTResponse]  ResponseCode: 200 |  |  |  |  |  |  |
| AP Payment | ResourceURI will have batch id which will be needed by Thirsty team to verify  /services/data/v58.0/jobs/ingest/7508N00000241i1QAA/ |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

* **(A) Service Billing Process:**
* TCCC\_ServiceBilling.WFL\_LoadGroupingID.workflow\_1
* TCCC\_ServiceBilling.WFL\_ThirstyCollect.Billing
  + Collects by making call to Thirsty and Stores in
  + SPLITTING\_DIRECTORY /usr/sap/interfaces/MZP/ServiceBilling/Incoming
* TCCC\_ServiceBilling.WFL\_SplitandFilterDupWO
  + Gets from above folder and send to COLLECTION\_DIRECTORY /usr/sap/interfaces/MZP/ServiceBilling/Incoming/Splitted
  + Checks if any duplicate in the Incoming folder
* TCCC\_ServiceBilling.WFL\_BP\_LookUp.FWD1
  + Collects from Splitted and sends to
    - BILLING\_COLLECT\_DIR

/usr/sap/interfaces/MZP/ServiceBilling/ToBilling

AND

* + - PAYSP\_Collecy\_dir
      * /usr/sap/interfaces/MZP/ServiceBilling/PaySP
    - For Survey WO (st=9/trc=73) – Only PaySP
* TCCC\_ServiceBilling.WFL\_PaySP.FWD1
  + TCCC\_ServiceBilling.WFL\_InvoiceStatus\_SplitFiles.workflow\_1
    - 50 rows
  + TCCC\_ServiceBilling.WFL\_InvoiceStatus\_BulkAPI.Process
    - BULK API (Jobs are created in CM Table)
    - “AP Acceptance”
* TCCC\_ServiceBilling.WFL\_CustomerBilling.FWD1
  + Picks from /usr/sap/interfaces/MZP/ServiceBilling/ToBilling and sends to
    - Warranty\_Out /usr/sap/interfaces/MZP/ServiceBilling/WarrantyClaims
  + TCCC\_ServiceBilling.WFL\_WarrantyClaims.Process
    - Picks up from above location
* **(B) Service Billing Process - Accrual**

1. **TCCC\_ServiceBilling.WFL\_LoadGroupingID.workflow\_1**

**Summary :** Read the file and load Group Id into a CM Table.

**Sample Input File :**

* 1. <add later>

**Workflow Steps and PseudoLogic :**

1. Collect (Disk)
   1. Directory : GROUP\_ID\_DIRECTORY : later
   2. Filename : GROUP\_ID\_FILENAME : later
2. Decode (Decoder)
   1. Put some code from ultra
3. Load Table (Analysis Agent)
   1. Consume Block :
      1. Delete the content from the DB Table
      2. Insert UDR into DB Table - GROUP\_ID
4. **TCCC\_ServiceBilling.WFL\_ThirstyCollect.Billing**

**And TCCC\_ServiceBilling.WFL\_LoadGroupingID.ThirstyCollect\_RT.Billing**

* **Sample File –** Not Applicable (Query from Thirsty)

**Workflow Steps :**

1. **GetAuditId (SQL) :**

Generates a unique Audit ID number for every run and pass it to next step.

1. **Start Batch (Analysis Agent) :**
   1. **Begin Batch :**
      1. Get query details (billing/accurals) – variant of the workflow
   2. **Consume Block :**
      1. Forward to RT workflow
2. **TCCC\_ServiceBilling.WFL\_LoadGroupingID.ThirstyCollect\_RT.Billing**
   1. **WFB\_From\_Batch (workflow Bridge)**
      1. Has UDR types of ConsumeCycle and WorkflowState
      2. Profile has response timeout of 60 sec ?
   2. **WFB\_Handler ( Analysis Agent)**
      1. Consume Block : just routing of udr’s to req or resp
   3. **Request SSN (Aggregation)**
      1. Aggregation Profile - PRF\_Agg\_ThirstyQuery
      2. SessionInit – map auditId, queryType, queryUrl to session object
      3. Consume –
         1. If udr is type QueryRequest, route to next step – ExecuteAuery
         2. If udr is type QueryResponse, retry to connect,
         3. if(session.queryType == "DSC")
            1. call parseDSCResponse

(understand this code )  
Query Type = DSC is used to query Thirsty for the current possible values for a “Detailed Service Code”. A DSC is a code string which describes the problem addressed by the workorder. It’s filled out by the service provider after a workorder has been completed.  
The code is complicated and consists of 5 parts, each of which might be 1-2 characters long.  
(Some values are funny: The Trouble Found field can have a code which means “Animal in equipment”)   
Sometimes there is padding with spaces, sometimes not. There are also rules for which combinations are allowed. The DSC query pulls all possible values from Thirsty so that CM doesn’t have to parse the current value. (I originally did parse it and it worked 99% of the time, but 1% failure is too much…)   
**I AM UNSURE WHETHER THIS IS ACTUALLY USED NOW**-- Stefan

* 1. ExecQuery : Analysis Agent
     1. Getting the Query String from an APL\_Query file and sending the udr to Thirsty Rest Client.
     2. For Billing, Where Clause – Invoice\_Status = Approved for AP
     3. For Accurals, Invoice Status = 'Reinstated','Hold', 'Pending Analyst Review', 'Disputed'
  2. SFDC\_REST
     1. Makes the call, response is in RestCycleUDR
     2. Has key parameters in External Reference
  3. ExecQuery : Analysis Agent
     1. If RestCycleUDR 🡪 Call createResponse - Generates response for different combinations and converts to QueryResponse UDR
  4. **Request SSN (Aggregation)**
     1. **If udr is** QueryResponse 🡪 Json response from rest call is converted to InvoiceResponse and then to WorkOrderResponse UDR and then sent to route ‘wo\_out’ for Billing Scenario
  5. **Transform**
     1. **If WorkorderSet udr , WorkOrders**
        1. **Put all workorders into an xml file**
        2. **Filename Format Workorders+AuditId.xml**
        3. **If another workorder has same outletacn present again , then add it to second file or create new file (CC limitation / Object locking issue)**
        4. **Filename Format Workorders+AuditId\_1.xml**
        5. **If again same outletacn is there, created \_2. \_3… and so on.**

**Sample xml file**

**<insert here>**

* 1. **1. WorkOrderOut**
     1. **Writes the xml to Collection\_Directory /usr/sap/…(do later)**

**2. EncodeNotif**

**Not used – code commented.**

* 1. **DSC Call Flow :**

**Not Completed – Do it later - Not a key flow.**

**Workflow - BP Look Up**

* All charges should match the total. If not Discard the Workorder.
* We do lookup checks from SAP/SOM for BP TYPE = OUTLET, then HQ , then Chain and get the Tech id ( I think its contract MA\_ID) . Here we map the TechId in the xml
* If we get error in retrieving Outlet TechID**,** Discard the Workorder. (Email is sent).
* But if we get error in Tech ID for HQ and Chain – we still continue.
* There is also a check if Customer Billing CIT’s or SP Payment Document for that WO are present in SAP.
  + If no errors, clone the WO udr and send to Customer Billing and PaySP.
  + If Customer Billing CIT’s are already present, We set a flag ‘FromDupCheck’. If above flag is true and also SP Payment Document exists, then still send it PaySP path and Also it goes to Invoice Status WFL for the backfeed.
  + If Customer Billing CIT’s are already present, We set a flag ‘FromDupCheck’ and PaySP doc is not created, the WO goes to PaySP.
  + If PaySP documents are already present and Customer Billing doc is not created the WO goes to Customer Billing.
  + For Survey WO ( TRC = 73 and Service Type – 9), if we get error in Customer Billing (Contract not found), then we send it to ‘Incoming’. This is done so that when a contract is created in SAP at a later date – it can be processed.

**Email Error - Error Log for Parts cost validation in Service Billing - SP0**

Example - For workorder [D175819711] cost is not matching, WO has overall cost of 1144.16 and cost breakdown is 0.0

**Customer Billing**

**WFL\_CustomerBilling (Batch WFL)**

**Do later – not much – pass workorder to RT**

**WFL\_CustomerBilling\_RT (Realtime Workflow)**

**Wrap\_WFB (Analysis Agent) :**

* We are getting the workorders from batch wfl
* We are making a BP look up call for BP Type = Outlet, and if SAP does not return a Valid Outlet TechID , then we will discard that Workorder. Note – we don’t need to map the Outlet TechID into the xml in this step as it was previously done in BP lookup workflow.

**Workorder\_Session (Aggregration Agent )**

* Session Key is Workorder Id.
* In Sesssion Init,
  + For ServiceBilling run and If one of the parts has warranty flag as true.
    - Setting invoice\_status = Warranty
    - We will create Warranty Claim xml for next workflow.
* In Consume Block,
  + If HQ and Chain TechID are not null and if not an Accrual Run,

then we make a SessionStartUDR call to SAP/BRIM and retrieve Free Call information for that WO.

* + - If the response of SessionStartUDR is successful, then all CIT’s created till we trigger SessionStopUDR will be applied with Free Call and it will be not chargeable to Customer.
    - Free Call is not applicable for Warranty WO (if one part has warr flag true) and for Accurals. Its only for WO in ‘Approved for AP’.

ELSE

Create Z100 CIT (Workorder CIT)

And here pass Accural Indicator (CC\_PROP\_ACCRUAL\_IND)– ACE if it’s a accrual run or else blank.

* + On Getting CC Response of SessionStartUDR- (instanceOf(input, CCWrapper))
    - * If the response for Chain Techid is having error, then send SessionStartUDR for HQ.
      * Check with Ray on this part of code.
      * Create Z100 CIT (Workorder CIT). And here pass Accrual Indicator (CC\_PROP\_ACCRUAL\_IND)– ACE if it’s a accrual run or else blank.
  + On Getting CC response of Z100 - instanceOf(ccwrap.request, ChargeUDR))
    - * Method warrantyAddTransactionDetails will add transaction details (maybe 4-5 rows – misc, labor…) coming from Z100 response and this will be written in WarrantyClaim xml ( it will create Z130 CITS in Warranty workflow )
      * If error in Z100,
        + create SessionStop UDR if we have a freecall (or meaning SessionStartUDR was success n retrieved freecall).
        + If no freecall, createResponse will do ….  
          IF there was no free call, there’s no session and nothing special needs to be done to close out anything. Still check createResponse, however. -- Stefan
      * If Accrual Run
        + Create Z100 CIT with AccInd = ACR.
        + Create Parts CIT (Z110) as many parts exist

Acc Ind = ‘ACR’

Add certain field only if warr flag = true

If not Accrual Run

* + - * + Only Create Parts CIT (Z110) as many parts exist

Acc Ind = ‘’

Add certain field only if warr flag = true

* + - * + Create DSC CIT(s) as many DSC code recd.
      * Create and send SessionStopUDR call