## DPI Interface

### Data Visualization

### Data population

### New Job data

* Must apply a business rule to determine if job data is required for extraction to the DPI
* **BR001**
* If a job record contains a job status of ‘Active’ and the status create date is “=” to the current date, then extract and store DPI data. Insert the record into the DPI dashboard table view of new jobs and flag as ‘New Jobs’, else if the status create date is “>” than the current date flag as ‘Continuing Jobs’ DPI data
* If the conditions of BR001 have been met, then extract job attribute data for display within the new jobs view
* **Job Data**
* Job number
* Division
* Primary division
* Working divisions
* Customer
* Job location
* Proposal # (if applicable)
* Derailment details
* # of engines
* # of loads
* # of empties
* Job action
* **Resources Data**
* Equipment
* Descriptor
* Unit #
* Combo #
* Division
* People
* Name
* employee ID
* Division
* position
* Once the job data has been determined the system must verify if the job is a multi-division job
* If job record contains 2 or more divisions, then group resource data with the given division at the job level
* Must run business rule to determine if corresponding call log data exist
* Run **BR002** to determine if DPI data contains any call log data with a log create data the is ‘=’ to the current date, if records exist run BR004 to verify if the start/stop times that need to be applied for the resources of the job
* Must provide an option to default user to ‘New Job’ DPI data for display within the data grid for viewing and/or processing within the dashboard view, as well as display an option to process ‘Continuing Jobs’ and provide a calendar widget to allow users to process date specific DPI data
* Must default the DPI dashboard view to automatically display and populate new job data based on the results of BR001 and BR004, without the user specifying a date range value

### Continuing job data

* Must apply a business rule to determine if extracted job data is required additional DPI processing
* **BR003**
* If BR001 is false, then determine if the active job has a status create date that is “<” than the current date, then flag the record as ‘Continuing Jobs’ and insert the record into the DPI table, else if the job status is no longer ‘Active’ wait for the DPI status of ‘DPI Done’
* If the conditions of BR003 have been met, then re-run BR002
* Run **BR002** to determine if DPI data contains any call log data with a log create data the is ‘=’ to the current date, if records exist run BR004 to verify if the start/stop times that need to be applied for the resources of the job
* Must provide an option to allows users to select “Continuing Jobs’ DPI data for display within the data grid for viewing and/or processing within the dashboard view as well as a calendar widget to allow the users to view and process date specific DPI data
* Must update the data grid and populate the requested data results based on the selection of ‘Continuing Jobs’ or date specific DPI data

### Start/Stop times

* Must apply a business rule to determine if extracted job data based on a true statement for BR001 and BR003 require corresponding call log data to be extracted, to determine accumulated work hours for the given resources of the job
* BR004
* Start time rules
* **BR001P - Initial start time rule**

If the job call date is = or < then the current date, then apply start time based on the earliest action time for call types of ‘Transfer’, ’Added Resources’, ‘Work time’ or ‘Mob’

* **BR001E - Emergency jobs start time rule**

If a job has an ‘Emergency response required’ flag and the job initial call date = or > than the current date, then apply the job initial call time as the start time for all resources assigned to the job, else if a call type of ‘Ordered’ exist after initial resources have been added or the current time is = or > 12mid, then apply start time to the additional resources based on the earliest action time for call types of ‘Transfer’, ’Added Resources’, ‘Work time’ or ‘Mob’

* End time rules
* **BR003G - End time rule**

If BR001P or BR001E exist and a value exist for SOC, SOR or Parked, then apply the end time to the start time of the given resource and calculate accumulated hours and the given end tracking cycle, else if run BR002I and flag resource as “INSF”

* **BR002I – Invoke call end time**

If no end time conditions exist then calculate from the start time till the time in which the DPI was called to sum the total work hours and flag as ‘INSF’ and flag for report disclaimer

* Must be able to auto insert a disclaimer when the generate report feature is called upon to advise the users that revenue is not based on actual start and stop times

“All records listed as INSF have assumed end times of <current date/time> for calculation purposes”

* Must store tracking cycles of a complete start/stop time occurrence and calculate the total number of hours per resource
* Must be able to sum the total occurrences of the calculated start/stop times and insert the ‘Total working hours

### Data Input

### Work Hours

### Auto-Insert value

* Must be able to insert the total work hours per resource based on the work hour logic rules
* Must set the inserted value in an editable mode

### Manually inserted value

* Must allow the users to override the inserted value
* Must set the formatting of the hours field to allow a number value with a double digit decimal place (5.75 = 5 hours and 45 minutes) to account for the total number of hours and minutes

### Flag indicator

* Must flag values that have been overwritten as ‘M’ for modified

### Reset feature

* Must provide a feature to allow users to reset the auto-generated value for the work hours if needed. Re-run logic rules to get the latest and greatest value.

### Continuing flag

* Must provide a feature to allow users to flag resources as continuing for items that are working throughout the night
* Must auto set resources flagged as continued to an end time of 12am
* Must not flag resource as INSF

### Rate Amount

### Auto-Insert value

* Must be able to insert the total work hours per resource based on the rate table logic rules
* Must set the inserted value in an editable mode

### Manually inserted value

* Must allow the users to override the inserted value
* Must set the formatting of the rate field to allow a currency value with no decimal placement

### Flag indicator

* Must flag values that have been overwritten as ‘M’ for modified

### Reset feature

* Must provide a feature to allow users to reset the auto-generated value for the rate if needed.

### Permits

### Manually-Inserted value

* Must allow the users to enter a value
* Must set the formatting of the permits field to allow a numeric value to specify the # of states which heavy equipment traveled through
* Must calculate the numeric value entered by $100
* Must require input data to be captured at the combo level if a combo exist on the job
* Must only assoicate meal cost to the resource type of equipment
* Automatically default the number of states to 1 at the combo level for the primary resource

### Meals

### Auto-generated value/Manually-Inserted value

* Must allow the users to select a value (shouldn’t this value be pre-populated based on the start/stop times to determine the # of hrs worked to determine the meal cost and allow the users to modified the value if needed?)
* Must set the formatting of the meals field to allow a numeric value to specify the # of hr per meal
* Must only assoicate meal cost to the resource type of people
* Must calculate the numeric value entered by the following:

6 hrs = $10 per meals

12 hrs = $20 per meals

18 hrs = $30 per meals

24 hrs = $40 per meals

* Must require input data to be captured at the resource level (people)
* Must provide the users with a feature to allow them to reset the auto-generated value if needed

### Hotel

### Manually-Inserted value

* Must allow the users to select a checkbox to flag the resources as ‘Hotel’, to specify the employee staying at a hotel
* Must set the checked hotel checkbox to populate $70 in the hotel cost field and allow the users to modify the value if needed
* Must only assoicate meal cost to the resource type of people
* Must provide the users with a feature to allow them to reset the auto-generated value if needed

### Process DPI

### DPI Dashboard view

### Manually triggered

* Must provide a way to allow users to process a DPI data at the job level
* Must be able to insert the total work hours per resource based on the work hour logic rules
* Must set the inserted value in an editable mode

### Default view

* Provide an accordion type view for the process DPI screen to show the published rates, special pricing and bid rates
* Auto-insert data from the job record if applicable and populate the special pricing data and auto-calculate based on input data
* Auto-insert data from the job record if applicable and populate the bid rate data and auto-calculate based on input data
* Always calculate all job data against to populate the published rate data
* Must provide a view that will always calculate and present job data at the published rate level regardless of the price type
* Must require users to select the price type revenue for final approval prior to saving the reporting

### Logic

### Trigger based

### Invoke DPI

* Must be able to automatically run the logic when the DPI function is called and determine if a start/stop time exist prior to running the calculations and populating a revenue value
* Must be able to run validation rules to determines if start/stop times apply to verify if INSF flags need to be displayed
* Must calculate up till the time in which the DPI function was called and insert the current date/time if no stop time exist, flag record as INSF

### Process DPI

* Must be able to present populated data upon invoking the function of the DPI
* Must be able to allow the users to select to process DPI data at the job level for new jobs, continuing jobs or date specific jobs

### Data extraction

### Primary Rate table

* Must be able to store and populate data from ProClarity that is extracted on a nightly basis from logic and calculation purposes
* Must be able to update and populate the primary rate table

### Rate tables

### System auto-generated rate table

* Extracted data from ProClarity must be stored and maintained within the primary rate table
* Data must be formatted as read only
* Must run a nightly service to refresh the database and mirror ProClarity
* Must be able to populate and display the source data within the DPI dashboard view in an updateable mode to allow the users to change the rate for DPI processing if needed
* Must be able to flag modified rate data
* Must allow users to reset modified rate data if needed, setting it back to the primary rate value

### User generated rate table

* Must provide a table structure that will allow users to enter and manage secondary rate data
* Provide an interface to capture and store data at the resource level by equipment and position type, published country rates, customers
* Must allow user to make edits to the data
* Must allow data to be overwritten without keeping a history or modified users
* Data must be formatted as updatable
* Must be able to populate and display the source data within the DPI dashboard view in an updateable mode to allow the users to change the rate for DPI processing if needed
* Must be able to flag modified rate data
* Must allow users to reset modified rate data if needed, setting it back to the secondary rate value
* Must be able to determine if the job record (Customer data) required for processing exist within the primary rate table
* If customer data is not found within the primary table, must run logic off of the secondary rate table
* Calculate the start/stop times at the current day if processing the new job records
* Calculate the start/stop times for all active continuing jobs for the current date/time for the earliest occurrence of a start/stop time
* Only display jobs from the previous day that are active from the new jobs

### Validation rules

### Work hours

* Must be able to determine the total number of work hours
* Must be able to determine if a stop time value does not exist and flag the entry as INSF
* Must be able to insert the total work hours per resource based on the work hour logic rules
* Must set the inserted value in an editable mode
* Must set the work hour value(s) in an editable mode after population from the rate tables logic to allow the user to modify the auto-inserted value if needed
* Must be able to flag modified rate data

### Auto-calculations

* Must provide an interface layer that will allow the users to view and modify data based on the work hours logic and rate logic
* Must be able to apply the work hour logic rules to determine the ext. rate calculated value
* Must be able to times the hours value by the rate value (primary or secondary rate value) to populate and display the ext. rate value in a read only mode

### Report Generation

### Approved job revenue

### Invoke Generate DPI Report function

* Must be build conditions to trigger the Generate DPI report, where the daily revenue for a specific data set (New jobs, continuing jobs, reprocessed jobs) are validated to ensure that all jobs have been processed prior to producing the final DPI report
* Must alert users of unprocessed job records that fail the validation process
* Must present a function on the main DPI dashboard page to allow users to request to generate the DPI report
* Must base the reporting data set based on the data view that has been processed

### Generate DPI Report

### Report layout

* Report Header
* Must be able to display header details for new jobs, labeling the report as “New Jobs – Daily Performance Indicator”

### Reporting logic

* White board data
  + 1. **Scheduled Presets** 
       1. **Triggering Event to extract data**
          1. As a user of the application we will only require a single condition that will require job related preset data to be extracted and dumped into DPI data view
          2. As a user of the application we will require job preset data for the requested date to be presented in the DPI data view
       2. **Data Extract Type** 
          1. As a user of the application we will require the following data to be extracted from job records and displayed in the DPI data view for viewing and exporting purposes
* Data Set
* Division
* Customer
* Location,
* Job description
* Reserved equipment type
* Quantity
* Reserved division
* Date
  + - 1. **Data presentation**
         1. As a user of the application we must be able to view job preset data
         2. As a user of the application we must be able to view job preset data as a result of the request date within the table view
         3. As a user of the application we must be able to export job preset data from the table view
         4. As a user of the application we must be able to print job preset data from the table view
         5. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “**Scheduled Presets**”, applicable columns(refer to the data set) and associated data results, within the DPI data view
    1. **“WON” Bids**
       1. **Triggering Event to extract data**
          1. As a user of the application we will only require a single condition that will require job related ‘Won’ bid data to be extracted and dumped into DPI data view
          2. As a user of the application we will require a business rule to determine if a bid has been ‘Won’
* **BR001** – If a Bid record has been converted into a job record, where the proposal # is linked to an non- job # and the job status for the requested date = ‘Preset’, then flag as a ‘Won’ bid and display in DPI data view
  + - * 1. As a user of the application we will require job ‘Won’ bid data for the requested date to be presented in the DPI data view
        2. As a user of the application we will require job related ‘Won’ bid data to be extracted based on the users request date entered into the “View DPI Data” function
      1. **Data Extract Type** 
         1. As a user of the application we will require the following data to be extracted from job records and displayed in the DPI data view for viewing and exporting purposes
* Data Set
* Division
* Customer
* Location,
* Job description
* Reserved equipment type
* Quantity
* Reserved division
* Date
  + - 1. **Data presentation** 
         1. As a user of the application we must be able to view data for Bid jobs that have been won
         2. As a user of the application we must be able to view data for Bid jobs that have been won as a result of the request date within the table view
         3. As a user of the application we must be able to export Bid job data that has been won from the table view
         4. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “”**WON” Bids**”, applicable columns(refer to the data set) and associated data results, within the DPI data view
    1. **DPI Offenders list** 
       1. **Triggering Event to extract data**
          1. As a user of the application we will only require two conditions that will require ‘INSF’ job related data to be extracted and dumped into DPI data view
          2. As a user the application we will require ‘INSF’ data for active jobs with the requested date to be presented in the DPI data view
          3. as a user of the application we will require a rule to run against the DPI data view ‘Date’ field
* **BR002** - If an ‘INSF’ flag exist and the ‘INSF’ flag create date is “= or >” the requested date field in the DPI data view, then do not populate the DPI offenders list

**Else,** if the “INSF’ flag create date is ‘<’ then the requested date field in the DPI data view, then run BR003

* **BR003** – If a continuing job flag exist, then do not populate the DPI offenders list

**Else,** then run BR004 and populate the DPI offenders list

* + - * 1. As a user of the application we will require DPI offenders list data to be extracted and populated into the table view if BR003 is “False”
      1. **Data Extract Type** 
         1. As a user of the application we will require the following data to be extracted from job records and displayed in the DPI data view for viewing and exporting purposes
* Data Set
* Job Number
* Division
* Name
* Customer
* Location
  + - * 1. As a user of the application we will require rules to determine which resource/name is associated to the offense
* **BR004** – If BR003 is ‘False’, then determine if a call type of ‘work time continuing’, ‘work time release’ or ‘demob’ exist and run BR005

**Else,** determine the last stored call typeand populate the DPI offenders list ‘Name’ with the stored value for ‘Called in by’

* **BR005** – If BR004 is ‘True’ then determine the last stored call type from the three and populate the DPI offenders list ‘Name’ with the stored value for ‘Called in by’
  + - 1. **Data presentation**
         1. As a user of the application we must be able to view data for jobs that do not call in shut off times
         2. As a user of the application we must be able to view data for DPI offenders as a result of the request date within the table view
         3. As a user of the application we must be able to export DPI offenders data from the table view
         4. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “**DPI Offenders List**”, applicable columns(refer to the data set) and associated data results, within the DPI data view
    1. **ETA Accuracy** 
       1. **Triggering Event to extract data**
          1. As a user of the application we will only a single condition to determine ETA % data to be extracted and dumped into DPI data view
          2. As a user the application we will require ETA % data for active jobs with the requested date to be presented in the DPI data view
          3. As a user of the application we will require that any job record flagged with a missed ‘ETA’ be feed/copied to a table that will be able to easily store values for reporting purposes on a daily basis to support the DPI function
          4. As a user of the application we will require a rule to run against the DPI data view ‘Date’ field
* **BR006** - If the requested date field in the DPI data view is ‘<’ than the current date, then determine if any “ETA” call entries exist for the requested date. If values exist, determine which are flagged as ‘Early ETA 15 min+’, ‘Early ETA 30 min+’, ‘Late ETA 15 min+’ or ‘Late ETA 30+” and calculate the % “divide the given amount by the total amount and then multiplying the answer by 100 to get the percentage of the given amount” and populate the DTD (Day to Date) column, MTD (Month to date) column with % data for the requested month and populate the YTD column for the current year and dump into the ETA Accuracy table then run BR008

**Else,** If BR006 is ‘False’run BR007

* **BR007** - If the requested date field in the DPI data view, is ‘= or >’ to the current date, then do not populate the ETA Accuracy list for the DTD column

**Else,** populatethe MTD column with % data for the requested month “divide the given amount for the month by the total amount and then multiplying the answer by 100 to get the percentage of the given amount” and populate the YTD column for the current year “divide the given amount for the year by the total amount and then multiplying the answer by 100 to get the percentage of the given amount”

* + - * 1. As a user the application we will require ETA % data to be extracted for the main ETA % data for a specific company “BNSF” for all active jobs with the requested date to be presented in the DPI data view
* **BR008** - If any of the missed eta results have a customer value of ‘BNSF’, then determine the % amounts for ‘Early ETA 15 min+’, ‘Early ETA 30 min+’, ‘Late ETA 15 min+’ or ‘Late ETA 30+” and calculate the % “divide the given amount by the total amount and then multiplying the answer by 100 to get the percentage of the given amount” and populate the DTD (Day to Date) column and dump into the BNSF ETA Accuracy table
  + - 1. **Data Extract Type** 
         1. As a user of the application we will require ETA data to be extracted from the call logs and stored in an ETA table to support the ETA accuracy reporting
* Data Set
* Call log
* Customer
* Missed ETA call entry create date
* Early ETA 15 min+
* Early ETA 30 min+
* Late ETA 15 min+
* Late ETA 30 min+
  + - * 1. As a user of the application we will require conditional data based on BR006 – BR007 to be calculated to determine % amounts
* Data Set
* Early ETA 15 min+
* Early ETA 30 min+
* Late ETA 15 min+
* Late ETA 30 min+
* Customer
  + - 1. **Data Presentation** 
         1. As a user of the application we must be able to view data for jobs that exceeded the ETA by 15 or 30 minutes
         2. As a user of the application we must be able to view data for ETA as a result of the request date within the table view
         3. As a user of the application we must be able to export ETA data from the table view
         4. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “**ETA Accuracy List**”, applicable columns(refer to the data set 1.1.5.2.2) and associated data results, within the DPI data view
    1. **Missed ETA**
       1. **Triggering Event to extract data**
          1. As a user of the application we will only require a single condition to determine Missed ETA data to be extracted and dumped into DPI data view
          2. As a user of the application we must be able to view missed eta data in real time.
* **BR009** - If the requested date field in the DPI data view is ‘=” to the missed eta flag create date or the Late ETA 15 min+ flag create date or the Late ETA 30 min+ flag create date, populate the ‘Missed ETAs’ table
  + - * 1. As a user of the application we will require job related Missed ETA data to be extracted based on the users request date entered into the “View DPI Data” function
      1. **Data Extract Type** 
         1. As a user of the application we will require Missed ETA data to be extracted from the call logs and stored in an ETA table to support the Missed ETA report
* Data Set
* Call log
* ETA call log
* ETA time
* ATA call log
* Actual time
* Missed ETA Reason
* Missed ETA Reason
* Late ETA 15 min+
* Late ETA 30 min+
* Job record
* Division
* Job #
* Division manager
* Project manager
* Customer
* Location
  + - 1. **Data Presentation** 
         1. As a user of the application we must be able to view missed ETA data
         2. As a user of the application we must be able to view data for Missed ETAs as a result of the request date within the table view
         3. As a user of the application we must be able to export Missed ETA data from the table view
         4. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “**Missed** **ETAs**”, applicable columns(refer to the data set) and associated data results, within the DPI data view
    1. **Personal Off Duty** 
       1. **Triggering Event to extract data**
          1. As a user of the application we will only require a single condition to determine employee off call data to be extracted and dumped into DPI data view
          2. As a user of the application we will require employee off call data to be extracted based on the users request date entered into the “View DPI Data” function
* **BR010** - If the requested date field in the DPI data view is contained within the off call date range value for ‘Off call from’ and ‘Off call to’ then populate the off call date,

**Else,** if thedate field in the DPI data view is ‘=’ to the current date, then run BR011

* **BR011** – If the current time stamp is “= or >” than the off call ‘return time’ value, then do not populate the off call table

**Else,** if the current time stamp is ‘<” than the off call ‘return time’ populate the off call table

* + - 1. **Data Extract Type** 
         1. As a user of the application we will require off call data to be extracted from the employee profile maintenance table
* Data Set
* Off Call
* Employee
* Off call from (Single value)
* Off call to (Single value)
* return time (Single value)
* Proxy (Multi-value)
  + - * 1. As a user of the application we will require the ability to determine the division ID for the off call data to classify personnel by department
* **BR012** – If BR010 is true or if BR001 is false, then determine
* **If the Division ID is ‘= or <’ than 90** feed off call data to the ‘DM-ADM’ section of the Personal Off Duty table
* **If the Division ID is ‘=’ 97 or 98** feed off call data to the ‘SDM’ section of the Personal Off Duty table
* **If the Division ID is ‘= or <’ than 100** feed off call data to the ‘RVP – GM – DTO CORP’ section of the Personal Off Duty table
* **If the Division ID is ‘=’ 109** feed off call data to the ‘Safety’ section of the Personal Off Duty table
* **If the Division ID is ‘=’ 112** feed off call data to the ‘Sales’ section of the Personal Off Duty table
  + - 1. **Data Presentation** 
         1. As a user of the application we must be able to view off call data
         2. As a user of the application we must be able to view data for off call as a result of the request date within the table view
         3. As a user of the application we must be able to export off call data from the table view
         4. As a user of the application we must be able to view off call data in a structural layout according to the given departments
         5. As a user of the application we must be able to easily identify the table view and the export data set, showing the table header “**Personal Off Duty**”, applicable columns(refer to the data set) and associated data results, within the DPI data view
* Multi-division jobs
* Must be able to determine if a job has more than one working division associated to it.
* Must be able to apply logic to determine the total revenue amount for all allocated resources of the given working division and sum the total division revenue
* Call type data
* ETA call type
* Single division job
* Must be able to determine if a call type of ETA, exist for the given job or resources of the job
* Must be able to determine the earliest action time recorded for the ETA and insert it into the ETA column on the DPI report
* If no value exist for ETA, must determine if a call type of work time exist
* Must be able to determine the action time recorded for the work time and insert it into the ETA column on the DPI report
* Multi-division job
* Must be able to determine if a call type of ETA, exist for more than one resource of the job
* Must be able to determine the earliest action time recorded for the ETA and insert it into the ETA column on the DPI report
* Must be able to assoicate the action time at the working division level of the DPI for the given job
* If no value exist for ETA, must determine if a call type of work time exist for any of the resources of the job
* Must be able to determine the earliest action time recorded for the work time and insert it into the ETA column on the DPI report
* ATA call type
* Single division job
* Must be able to determine if a call type of ATA, exist for the given job or resources of the job
* Must be able to determine the earliest action time recorded for the ATA and insert it into the ATA column on the DPI report
* If no value exist for ATA, must determine if a call type of work time exist
* Must be able to determine the action time recorded for the work time and insert it into the ATA column on the DPI report
* Multi-division job
* Must be able to determine if a call type of ATA, exist for more than one resource of the job
* Must be able to determine the earliest action time recorded for the ATA and insert it into the ATA column on the DPI report
* Must be able to assoicate the action time at the working division level of the DPI for the given job
* If no value exist for ATA, must determine if a call type of work time exist for any of the resources of the job
* Must be able to determine the earliest action time recorded for the work time and insert it into the ATA column on the DPI report

### Reporting data

* Data dump
* Must be able to determine the DPI flag of a job record and insert job records flagged as ‘New Jobs’ to the new jobs report page
* Must be able to determine the DPI flag of a job record and insert job records flagged as ‘Continuing Jobs’ to the continuing jobs report page
* Must be able to insert DPI related data based on established business rules for the white board data
* Job number data
* Insert job number as the primary data element within the job record, this needs to be the highest level in the data set
* Pass data value from the logic for the requested DPI data set
* Division data
* Must be able to display all working divisions of a given job, associating the primary division at the job record line item level and displaying all associated working division under the job record line item, summing the division revenue total for all allocated resources of the given working division
* DPI Status data
* Must allow the final DPI report to print with INSF and flag the applicable records as such
* Must flag records that are identified as ‘Con.’ as continuing jobs on the new jobs report
* Must flag records that are identified as ‘Done’ as completed jobs on the new jobs report
* Must flag records that are identified as ‘Con.’ as continuing jobs on the new jobs report

### Print to PDF

### Invoke DPI

* Must be able to automatically run the logic when the DPI function is called and determine if a start/stop time exist prior to running the calculations and populating a revenue value