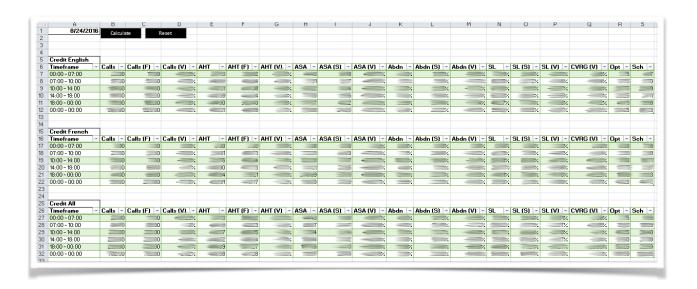# Intraday Stats Report

Summary

In December 2015, The Tangerine call centre began the "My Evolution" (MEVO) phase and moved from the Aspect WFM platform to the Genesys WFM platform. At this point in time, a lot of the reporting frameworks that were used by the IDA and Workforce teams became invalid due to the incompatibilities in the raw data structures and workflow processes of the two systems.

In the first half of 2016, I began working on the Intraday Stats Report. The main purpose of this report is to view section level summaries of various stats that determine whether we met our Key Performance Indicators (KPIs) and the reasons for why we did or did not meet them. Intraday Stats can be granular down to a 15 minute resolution and grouped into individual activities or a sub grouping of activities. The purpose of this report is to act as a feeder report to higher level reports and hence was designed with a report developer/designer to be its consumer.

Features

The stats that are present in this report are

- Call Volume (Forecasted, Actual and Variance)
- Average Handle Time (AHT: Forecasted, Actual and Variance)
- Average Speed of Answer (ASA: Forecasted, Actual and Variance)
- Abandoned Rate (Forecasted, Actual and Variance)
- Service Level (SL: Forecasted, Actual and Variance)
- Optimal Hours Required for Actual Call Volume
- Scheduled Hours available for Actual Call Volume
- Coverage Variance between Optimal and Scheduled for Actual Call Volume
- Calculated Hours Required for Forecasted Call Volume
- Scheduled Hours available for Forecasted Call Volume
- Coverage Variance between Calculated and Scheduled for Forecasted Call Volume

Other features of this report include
- High performance using the Range data structures in Excel through VBA to filter data extremely fast
- Easy to use with user interaction to generate reports is as simple as pasting raw data
- Flexible design to generate stats for multiple days to show patterns where performance is easily met vs where performance is suffering
- Able to generate stats at resolutions of15 minute intervals
- This report can be used for data visualizations which is useful at pattern recognition and determining scheduling inefficiencies
- A "Settings" feature where we can change activity grouping settings allowing us to drill down into individual activity stats
- Helper controls to reset and clear raw data when generating up to date stats

Evolution

The multi dimensional nature of the raw data in this instance meant that the sanitizing of the data including header columns was critical. Data sanitization played in an important part in the reduction of the complexity of the code that summarized the stats

Data filtering was done with the "Range" data structure in excel which provide substantial performance improvements. A lot of repetitive code was factored out to help with code maintenance.

Since this report is run at regular intervals throughout the day, time plays an important factor in the code and design. The code looks at the current time of day and then decides on the timeframes for stat calculations. Code snippets below show some of the code that was involved in stat calculation, data filtering and time based logic

```
For i = LBound(volumeArray) To UBound(volumeArray)
    sumWeight = sumWeight + volumeArray(i)
    sumWeightFC = sumWeightFC + volumeFCArray(i)
    sumProductASA = sumProductASA + (asaArray(i) * volumeArray(i))
    sumProductASAFC = sumProductASAFC + (asaFCArray(i) * volumeFCArray(i))
    sumProductAHT = sumProductAHT + (ahtArray(i) * volumeArray(i))
    sumProductAHTFC = sumProductAHTFC + (ahtFCArray(i) * volumeFCArray(i))
    sumProductSL = sumProductSL + (slArray(i) * volumeArray(i))
    sumProductSLFC = sumProductSLFC + (slFCArray(i) * volumeFCArray(i))
    sumProductABDN = sumProductABDN + (abdnArray(i) * volumeArray(i))
    sumProductABDNFC = sumProductABDNFC + (abdnFCArray(i) * volumeFCArray(i))
    sumProductCVRGVAR = sumProductCVRGVAR + (cvrgVARArray(i) * volumeFCArray(i))
Next

totalCalls = sumWeight
totalCallsFC = sumWeightFC
If sumWeight <> 0 Then
    weightedAverageASA = sumProductASA / sumWeight
    weightedAverageAHT = sumProductAHT / sumWeight
    weightedAverageSL = (sumProductSL / sumWeight) / 100
    weightedAverageABDN = (sumProductABDN / sumWeight) / 100
End If

If sumWeightFC <> 0 Then
    weightedAverageAHTFC = sumProductAHTFC / sumWeightFC
    weightedAverageASAFC = sumProductASAFC / sumWeightFC
    weightedAverageSLFC = (sumProductSLFC / sumWeightFC) / 100
    weightedAverageABDNFC = (sumProductABDNFC / sumWeightFC) / 100
    weightedAverageCVRGVAR = sumProductCVRGVAR / sumWeightFC

    If totalCalls <> 0 And weightedAverageAHTFC <> 0 Then
        volumeVariance = (totalCalls / totalCallsFC) - 1
        ahtVariance = (weightedAverageAHT / weightedAverageAHTFC) - 1
        If weightedAverageASAFC <> 0 Then
            asaVariance = (weightedAverageASA / weightedAverageASAFC) - 1
        End If
        If weightedAverageSLFC <> 0 Then
            slVariance = (weightedAverageSL / weightedAverageSLFC) - 1
        End If
        If weightedAverageABDNFC <> 0 Then
            abdnVariance = (weightedAverageABDN / weightedAverageABDNFC) - 1
        End If
    End If
End If
```

```
With ws.ListObjects(rawDataTable)
    .Range.AutoFilter Field:=3, Criteria1:=groupArray, Operator:=xlFilterValues
    .Range.AutoFilter Field:=4, Criteria1:=">=" & startTime, Criteria2:="<" & endTime, Operator:=xlAnd

    For Each rng In .ListColumns(volumeActColumn).DataBodyRange.SpecialCells(xlCellTypeVisible)
        volumeArray(UBound(volumeArray)) = rng.Cells(1).Value
        ReDim Preserve volumeArray(0 To UBound(volumeArray) + 1)
    Next

    For Each rng In .ListColumns(volumeFCColumn).DataBodyRange.SpecialCells(xlCellTypeVisible)
        volumeFCArray(UBound(volumeFCArray)) = rng.Cells(1).Value
        ReDim Preserve volumeFCArray(0 To UBound(volumeFCArray) + 1)
    Next

    For Each rng In .ListColumns(asaActColumn).DataBodyRange.SpecialCells(xlCellTypeVisible)
        asaArray(UBound(asaArray)) = rng.Cells(1).Value
        ReDim Preserve asaArray(0 To UBound(asaArray) + 1)
    Next
```
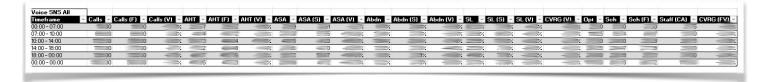
```
If reportDay = Date Then
    If Time > sevenAm Then
        timeString = "00:00 - 07:00"
        ThisWorkbook.CalculateSummary rawPerformanceTable, summaryTable, activityGroupingTable, mappingName, startOfDay, sevenAm,
timeString
    End If
    If Time > tenAm Then
        timeString = "07:00 - 10:00"
        ThisWorkbook.CalculateSummary rawPerformanceTable, summaryTable, activityGroupingTable, mappingName, sevenAm, tenAm,
timeString
    End If
    If Time > twoPm Then
        timeString = "10:00 - 14:00"
        ThisWorkbook.CalculateSummary rawPerformanceTable, summaryTable, activityGroupingTable, mappingName, tenAm, twoPm,
timeString
    End If
    If Time > sixPm Then
        timeString = "14:00 - 18:00"
        ThisWorkbook.CalculateSummary rawPerformanceTable, summaryTable, activityGroupingTable, mappingName, twoPm, sixPm,
timeString
    End If
    If Time > midnight Then
        timeString = "18:00 - 00:00"
        ThisWorkbook.CalculateSummary rawPerformanceTable, summaryTable, activityGroupingTable, mappingName, sixPm, midnight,
timeString
    End If
```

Because of the substantial amount of code in this report, the naming of variables and functions were very descriptive to allow easier code maintainability.

Nature of this Report

The Intraday Stats report plays an important role in that it is only a feeder report in the data pipeline to more higher level reports. The Intraday Service Status Report[1] is one example that gets its data input

from the Intraday Stats report. The design of the results that this report generates is meant more as a report developer view rather than an end user view.

---

[1] http://www.nowlabs.net/files/resume/Intraday_Service_Status_Report.pdf