**Driver Login calls DriverLoginProcess, PreferencesProcess, CodeTableProcess, CommodityMasterProcess, ContainerChangeProcess, TerminalChangeProcess, TripInfoProcess**

**Service: DriverLoginProcess**

Status: Fully functional

Limitations: Code has not yet been written to send GPS info to tracker.

Required Fields: EmployeeId,Password,PowerId,Odometer,LoginDateTime, PndVer

Special Fields: OverrideFlag = Y if driver has double checked Odometer and is resubmitting

?? Fields: LocaleCode, MdtId

Fields returned: TermId, RegionId, AreaId, TripNumber, TripSegNumber, DriverStatus

**Service: PreferencesProcess**

Status: Fully functional

**Service: CodeTableProcess**

Status: Fully functional

**Service: CommodityMasterProcess**

Status: Fully functional

**Service: ContainerChangeProcess**

Status: Functional

Limitations: Code has not yet been written to send entire container list if no ContainerMasterDate is received.

Special Fields: LastContainerMasterUpdate, populated to receive latest changes

LastContainerMasterUpdate , not populated to receive all containers.

**Service: TerminalChangeProcess**

Status: Functional

Limitations: Code has not yet been written to send entire terminal list if no TerminalMasterDate is received

Special Fields: LastTerminalMasterUpdate, populated to receive latest changes

LastTerminalMasterUpdate , not populated to receive all terminals.

**Service: TripInfoProcess**

Status: Functional

Open Issues: Need to be able to determine if we are sending trips because of a login request or because a new trip has been entered or an existing one modified for a driver that is logged in. Currently set to process login request.

**Driver Trip Acknowledgement calls DriverTripAck for each trip.**

**Service: DriverTripAckProcess**

Status: Fully functional

**Load/DropContainer action calls DriverContainerActionProcess**

**Service: DriverContainerActionProcess**

Status: Functional

Special Fields: ActionType: L=Load, S=Dropped

MethodOfEntry: M=Manual, S=Scanned.

**Enroute action calls DriverEnrouteProcess**

**Service: DriverEnrouteProcess**

Status: Functional

Limitations: Code has not yet been written to send scale notice for driver returning to a yard with a commodity on a pickup or load.

Code has not yet been written to resequence trips for driver if driver has gone out-of-sequence.

Code has not yet been written to send GPS Customer Host Code packet to tracker.

Open Issues: Delete TripSegmentMileage throws an exception, commented out.

Used only if a driver has previously gone enroute on this same segment.

**Arrive action calls DriverArriveProcess**

**Service: DriverArriveProcess**

Status: Fully functional

**Segment Done (Stop Transaction, Yard/Scale, Scale) calls DriverContainerActionProcess for each container and DriverSegmentDoneProcess for the segment completed.**

**Service: DriverContainerActionProcess**

Status: Functional

Limitations: Code to process the “quantity in container number” has not been written yet.

Testing needs to be completed for setting of the location warning flag.

Testing needs to be completed for exception and review processing.

Special Fields: ActionType: D=Done, E=Exception, R=Review

ActionCode: For Yard, Review Reason Code or Exception Code

ActionDesc: For Yard, Review Reason Description or Exception Description

SetInYardFlag: For Yard, Y/N

**Service: DriverSegmentDoneProcess**

Status: Functional

Limitations: Testing needs to be completed for exception and review processing.

Testing needs to be completed for scale segment trip type processing.

Open Issues: Delete TripSegmentContainer throws an exception, commented out.

Used to remove any additional containers that have not been completed from the trip. This should not normally happen.

**This service includes the MarkTripDone processing.**

Status: Functional

Limitations: Testing needs to be completed for exception and review processing.

Code has not yet been written to check for exception delay type. If any delay on the trip was an exception type, change the status of the segment to E=Exception.

Code has not yet been written to check if trip should be placed in the error queue. This is a preference.

Code has not yet been written to re-sequence trips or the driver, now that one has been completed.

**Driver Delay and Back on Duty Actions calls DriverDelayProcess**

**Service: DriverDelayProcess**

Special Fields: ActionType = D=Delay, B=Back On Duty

Status: Functional

Open Issues: Delete DriverDelay throws an exception, commented out

Used only when there are multiple open-ended delays. This should not happen.

Other issues: What do we do if there is no driver status record because driver is not logged in.

This should never happen. But should we create one and add a driver history record?

On a back on duty, if there is no delay record started, should we start one and end it?

**Driver Fuel Entry calls: DriverFuelEntryProcess**

**Service: DriverFuelEntryProcess**

Status: Fully functional

Open Issues: None

**Driver State Line Crossing calls: DriverStateLineProcess**

**Service: DriverStateLineProcess**

Status: Fully functional

Open Issues: None

**Driver New Container: calls DriverNewContainerProcess**

**Service: DriverNewContainerProcess**

Status: Fully functional when Container Number has not changed.

Status: Partially functional when Container Number has changed.

Open Issues: Delete ContainerHistory throws an exception, commented out.

This is needed to remove the old container number history and insert the new container number history, since we cannot just update the container number as it is part of the primary key.

Special Field: ContainerNumber, always populate even if it was not changed.

**Driver Sends a Message: calls DriverMessageProcess**

**Service: DriverMessageProcess**

Status: Fully functional

Fields: SendId field and other fields are populated.

Open Issues: None

**Driver Request Messages: calls DriverMessageProcess**

**Service: DriverMessageProcess**

Status: Fully functional

Fields: SendId field only field populated.

Open Issues: None