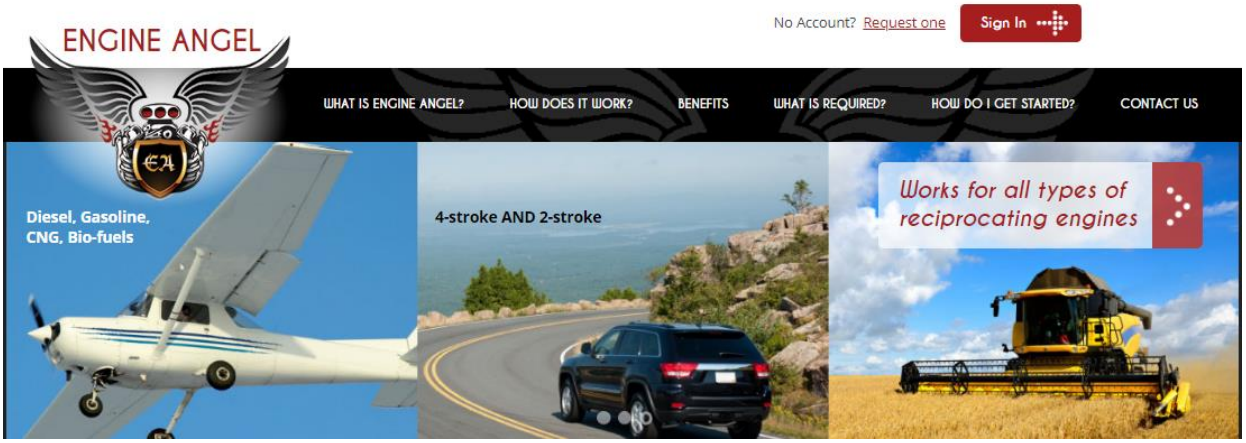
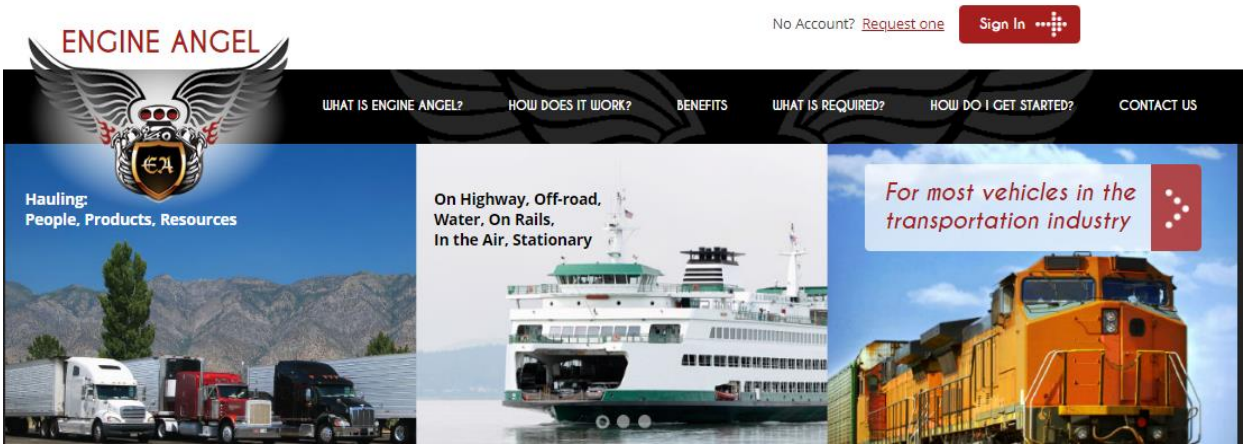
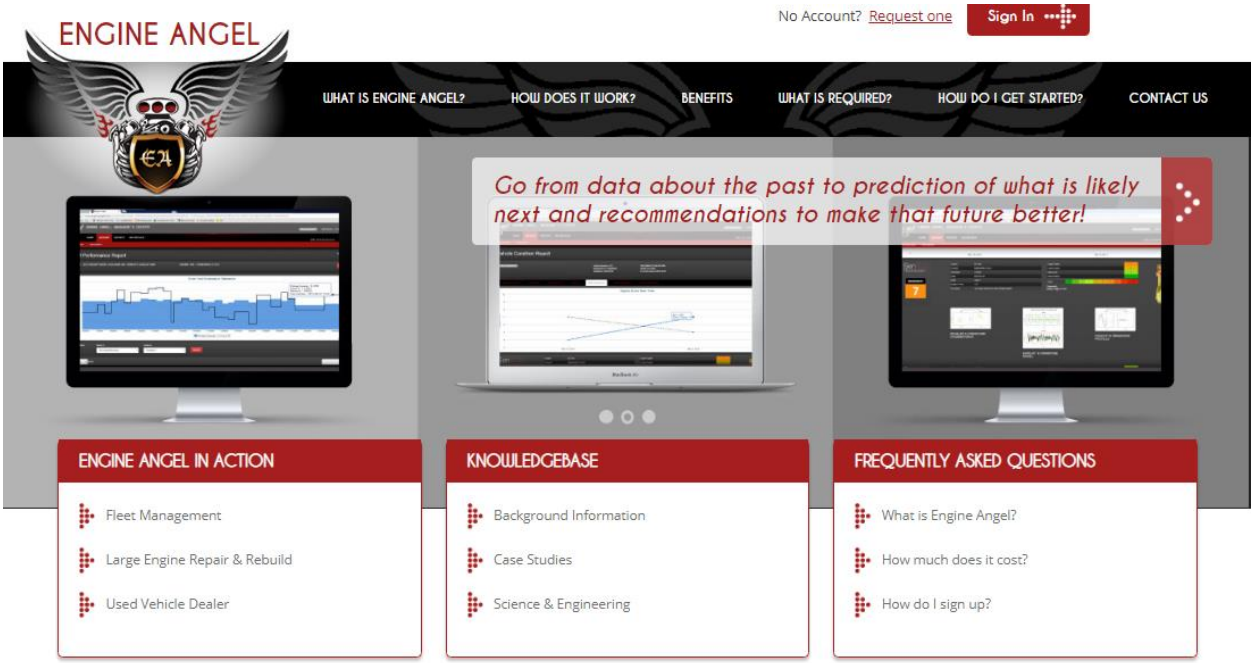
Engine Angel – Remote Data Sources







Engine Angel Documentation

Volume I. Admin Layer

2019-05-07

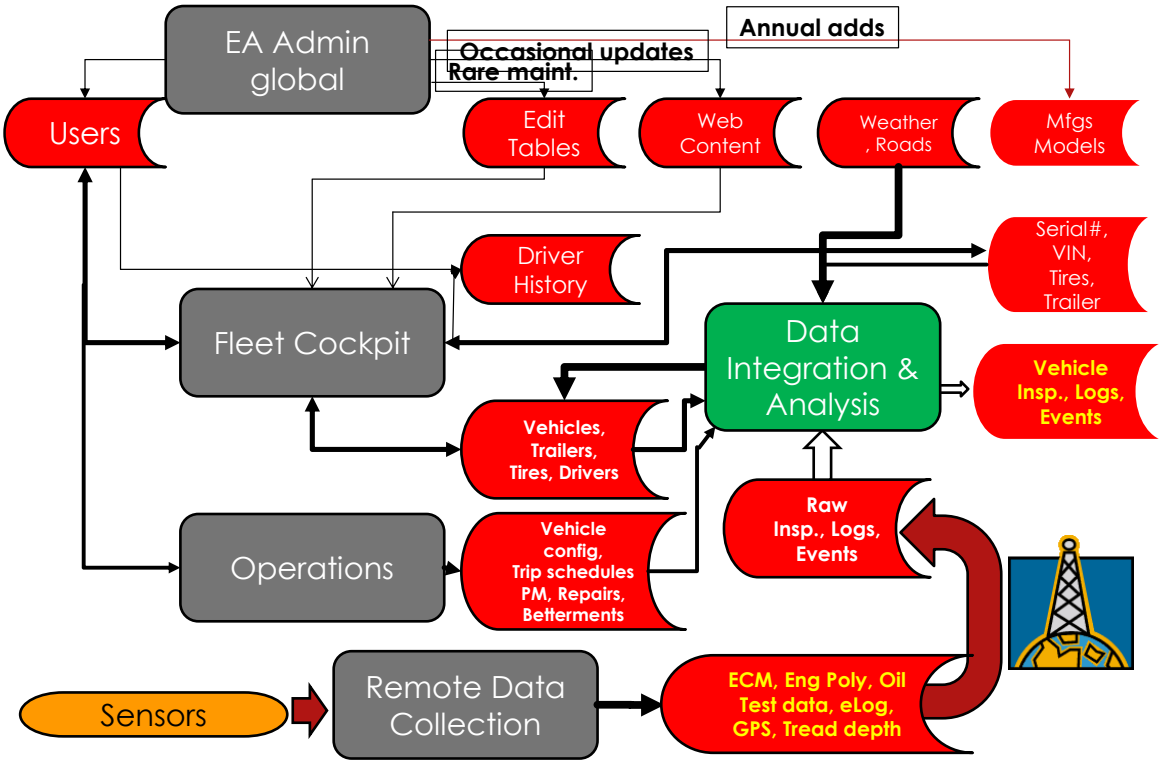
Contents

Introduction

The documentation for Engine Angel is provided in 5 volumes”

1. **Admin Layer (This document)** The **Admin** Layer for the staff of Predictive Fleet Technologies are responsible for much data that is independent of any customer Fleets (equipment specifications, conversion standards, regulations, industry standards, Expert System rules, etc.
2. The Fleet **Cockpit** for managers, professional staff responsible for the policy and management of operational people to achieve the organizations goals (managers, analysts, clerical people)
3. The **Operations** people who implement the actions according to policies and guidelines of management resulting in organizational success.
4. The **Remote** functions are those processes that connect sensors and special purpose accumulators of ‘short-term’ histories of data and transmit, transform and integrate the data into the system’s databases to keep data about asset status and activities accurate and timely.
5. The **Scheduled** functions (data integration and analysis) are processes that monitor asset status data against preset threshold values to alert operations and staff personnel of abnormal situations that might need corrective action. In addition, the batch processes use the historical data and models to predict future statuses against the preset thresholds to again alert and, using Expert Systems, suggest corrective actions to avert problems.

The following diagram (flow chart) represents these functions and major flows of data through the system.



Engine Angel Standards

Application Environment

The application is developed in PhP under the Microsoft .Net environment.

The EngineAngel homepage is developed under WordPress.

The database management system is MSSQL.

HIghCharts (JavaScript) is used for graphic displays of data.

PC components are developed in C##.

Change Control is managed through Team City software.

The production environments are:

EnginePolygraph.com and **EngineAngel.com**

It is occasionally rebuilt upon request from the QA environments.

The development environment consists of three components:

EP-DEV.GLTaC.com/Current and EA-DEV.GLTaC.com/Current

This environment is used by developers.

EP-DEV.GLTaC.com/Nightly and EA-DEV.GLTaC.com/Nightly

This environment is used by testers. The software is updated every early morning from Current

EP-DEV.GLTaC.com/QA and EA-DEV.GLTaC.com/QA

This environment is used by for final integrated testing. The software is updated periodically from Nightly.

Credit card payments are made using Recurly.

ETL uses SSRS to transform and update SQL tables from Remote inputs (files).

Alerts are batch programs the report through SSRS.

Advanced Analytics are developed in R and results reported via csv files.

Various Daily and Monthly batch jobs provide periodic summaries and data maintenance functions controlled by the MS scheduling software.

Authorization

All authorized UserNames are assigned to either the Admin Layer (Predictive Fleet Technologies employees) or the Cockpit/Customer Layers (Customer employees). Once authorized to access a Layer, the UserName is allowed to view all data, except personal data of other people and data under the Security tab.

Authorization to update data is assigned to a UserName by assigning the UserName the corresponding Role (Authorization) on the 'All People' table in the Admin Layer, or the Role using the Cockpit > Master Data > People table for the Cockpit and Operations Layers.

One UserName can access only the Admin or the Cockpit and Operations layers of EngineAngel, not both.

Application and Database table design

Database tables will have a LastUpdateUserName & LastUpdateDateTime to record last change/create source.

Tables have a 'list screen' with a string search function for key data from each row (entry).

The Export CSV button in the upper right causes the contents of the table of the list screen to be downloaded to your PC as a CSV file for your own reporting.

At the right of the screen, a pencil might be 'clicked' to get a detail screen for that row showing existing data and, if the user is authorized appropriately, changes can be made and the changes saved.

In many cases, the line will terminate with a bold, reddish '**X**'. If it is clicked, you will be prompted if you really want the row deleted. If you are authorized, clicking on 'yes' will cause the row to be deleted if it can be within the structure rules of the database (other things depend on it).

List screens also have bold reddish '**+**' in the upper right to indicate that you want to add a new entry into the table.

Some List screens end the item line with a 'additional information' icon  that will cause a branch to a screen of information related to the subject of he selected row.

Any session can be 'split' into two sessions by clicking on the white **+** in the red box on the upper right of each screen. This allows each session to have a thread in either of two tabs of your browser.

Help Screens are to be provided for each screen for the User to understand the purpose and general features of the data and business process involved.

Locale-specific data presentation

Event DateTimes are to be displayed in Fleet TimeZone; system DateTime is in UTC (as shown on the screens near the top right. Format is yyyy-mm-dd hh:mm.sss (I.e., 2017-01-25 22:07.843). In the csv files, the Date and Time are separated by an '_' to prevent Excel rules from erroneous interpretation.

Date input fields to have a Calendar pull-down in a popup window

Units of Measurement are to be metric in the Admin layer and as specified in the Fleet configuration data of Cockpit > Manage Fleet(s) > IsMetric (Y/N). Specific data elements can be defined in the Admin > Localization > UoM Data Rules. Conversions between various UoM are to be calculated using standard routines (SP) from data stored in the UoM Data table.

Distance calculations from GPS coordinated are to use the **Haversine method.**

GPS coordinates are to be stored in the DD (Decimal Degrees) format
Longitude values West of the Greenwich median are stored as negative numbers and positive for East of the meridian. Values must be between -180 and +180 so that if less than -180, add 360; if over 180, subtract 360.

Latitude values are between 0 (equator) and 90 at the North pole;
-90 is the South pole. If a latitude is >90, subtract 90; ; if less than -90, add 90.

Altitude is to be in meters (m).

To convert from DD to DMS (degrees, minutes, seconds):

Function Convert_Degree(Decimal_Deg) As Variant With Application

```
'Set degree to Integer of Argument Passed Degrees = Int(Decimal_Deg)
'Set minutes to 60 times the number to the right of the decimal for the variable Decimal_Deg
Minutes = (Decimal_Deg - Degrees) * 60
'Set seconds to 60 times the number to the right of the 'decimal for the variable Minute Seconds
= Format(((Minutes - Int(Minutes)) * 60), "0")
'Returns the Result of degree conversion (for example, 10.46 = 10° 27 ' 36")
Convert_Degree = " " & Degrees & "° " & Int(Minutes) & "' " _ & Seconds + Chr(34)
End With
End Function
```

http://support.microsoft.com/kb/213449

To convert from DMS to DD:

Function Convert_Decimal(Degree_Deg As String) As Double

' Declare the variables to be double precision floating-point. Dim degrees As Double Dim minutes As Double Dim seconds As Double

```
' Set degree to value before "°" of Argument Passed. degrees = Val(Left(Degree_Deg, InStr(1, Degree_Deg, "°") - 1))

' Set minutes to the value between the "°" and the "'“  of the text string for the variable Degree_Deg divided by 60. The Val function converts the text string to a number.
minutes = Val(Mid(Degree_Deg, InStr(1, Degree_Deg, "°") + 2, _ InStr(1, Degree_Deg, "'") - InStr(1, Degree_Deg, _ "°") - 2)) / 60

' Set seconds to the number to the right of "'" that is converted to a value and then divided by 3600.
seconds = Val(Mid(Degree_Deg, InStr(1, Degree_Deg, "'") + _ 2, Len(Degree_Deg) - InStr(1, Degree_Deg, "'") - 2)) _ / 3600

Convert_Decimal = degrees + minutes + seconds

End Function
```

There are three formats on the sidewalls of tires (**Tire size**):

wwIdd.d → ww is the width in inches. Convert to metric (cm) for database.
I=internal construction=[R,D,B], usually R=Radial
dd.d = diameter of wheel, normally to .5”. Convert to cm for database.
Assume aspect ratio = .9 so section height = .9*ww

www/aaIdd.d → www is width in mm. Convert to cm by dividing by 10;
aa = aspect ratio → section height = (aa/100)*width;
dd.d and ‘I’ are like above.

www/aaIddd → www is width in mm. Convert to cm by dividing by 10. Section height = (aa/100)*width;
ddd = diameter of wheel in mm, ,convert to cm for DB by divide by 10.

Overall diameter = wheel diameter + 2*(Section height)

7.50R16, 245/75R16, 190/65R390 are examples of each

Time Intervals

The human display for time intervals is of format: dd.hh:mm:ss (dd=days, hh = hours, mm = minutes, ss = seconds).

Time interval values are stored in db as seconds (s)

To go from database seconds (dbs) to human display,

dd = Floor(dbs/86400)

```
hh = Floor((dbs-dd*86400)/3600)
mm = Floor((dbs-dd*86400-hh*3600)/60)
ss = Floor(dbs-dd*86400-hh*3600-mm*60)
s = dd*86400+hh*3600+mm*60+ss
```

There are a number of components required for supporting multiple languages in Engine Angel.

Multiple Language Support

The UTF-8 codepage is required in the various text fields to support the various character sets, and ideograms of the various languages of the world.

Language codes are the 2-character, lower case ISO codes.

Since the Admin Layer is only for PTF access, it is designed for users with English literals and messages. However, data is to be visible in any selected language. Adds, changes, and deletes of codes can only be made in English since that is the language code that will update relationships between codes. Translations are managed by separate, usually batch, functions.

A language must be specified for the fleet which will be the language used for all batch reports.

The user may identify a different language for display of data at the time of sign-on (with ability to change during the session). This will enable the system to display standard descriptions, field titles, and error messages in the language requested. Comments made by other members of the fleet should be made and will be displayed in the fleet language.

Field titles and literal texts provided by the GUI are maintained in a ML-Contents table in the various supported languages.

The CodeLists table identifies codes in their relationships in a set of tables maintained in the Admin layer under the classification menu. The translations of those codes, even into English, can be provided with the batch functions in the translations menu of the admin layer. They are stored in ML- extension tables. The CodeLists and Codes can only be added, deleted or changed only in English when using the online screens; non_English Codes and Descriptions can be viewed in the Admin layer by changing the language (Locale) selection.

A number of tables contain language specific descriptions of groups and objects. These tables have ML-extensions for the corresponding translations. The maintenance of the translations are provided by requested downloads of the English content with the corresponding local language content; these tables are sent to translators, updated, and returned for upload into the ML- extension tables.

Field edit messages in maintenance programs are extracted from the programs and stored in the 'messages' table. ML-extensions provide the local language equivalent after translation.

Texts associated with Alerts, Work Orders, and other batch generated reports for users are maintained in the admin layer under the Alerts/Contents menu.

The following pages in this document detail specific features of the MultiLingual capabilities.

ADMIN => Localization => Languages => (below list screen)

EngineAngel > Translation Foundation > Enabling Engine Angel Translation.pptx

EngineAngel > Translation Foundation > ML Table Maintenance.pptx

EngineAngel > Translation Foundation > MultiLingual Application Messages.pptx

EngineAngel > Translation Foundation > CodeList Tasks.pptx

EngineAngel > Translation Foundation > Translation Tools for Translators.pptx

Engine Angel Data Layers (Flows)

The Engine Angel Application functions can be grouped by Interactive Functions, Remote Functions (such as data collection and integration from sensors and special purpose computers) and scheduled (batch) functions to analyze data for status changes and predictions to generate Alerts to people to consider 'corrective' actions.

Remote Data

ECM

ECM Synchronization

ECMDataSync

Engine Polygraph

EPReader

EPDataSync

EngPolygraph ETL to Engine Angel

Engine Oil Test Results

OilDataSync

Oil Test Results Synchronization

Telematics

Forecasting and Prediction

We use the term forecasting when we estimate a future expectation based on a timeseries analysis of previous activity by the subject asset. The timeseries analysis may use seasonal factors for adjustments based on the geographic location of the home base of the asset.

We use the term prediction when we use external variables to calculate a future expectation based on equations obtained from calibration with facts from the past for that asset.

Methods

Exponential Smoothing

Regression

Seasonal Factors

Hazard Functions

Bathtub Curves

Processes

Each component that is as specified by a work order template and assigned to an asset has details associated with the lifecycle of the asset and the deterioration models that will be used to predict eventual end of life (EoL) dates that will be used to prioritize and schedule shop actions. Initial deterioration rates are largely derived from complement warranties. It can be modified by the vocation assigned to the asset of the component.

Overview

All components to be managed by work orders require initialization of needed deterioration equations at the time that the component is brought into service.

As the asset is used, compliments attached to the asset will have utilization factors assigned along with hazard factors associated with the duty during that utilization. The deterioration equation will convert those utilization and hazard factors to deterioration rates. Forecasted utilization will be applied to those rates to predict deterioration in the future.

The future deterioration will be compared with the current component status and a threshold value that defines the EoL.

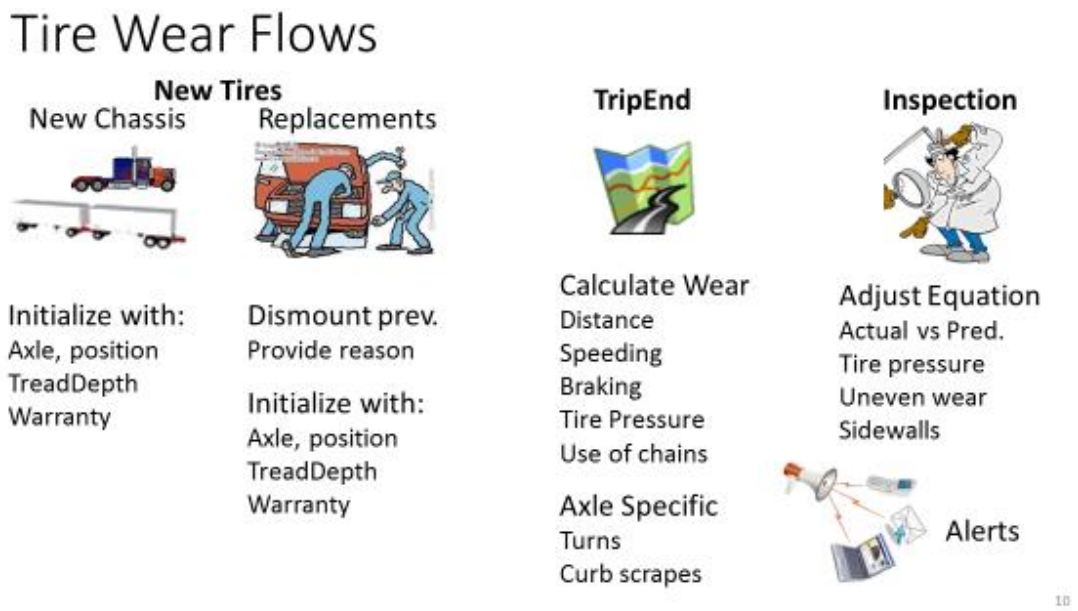
Actual utilization and hazard functions are extracted from the ECM and used to predict deterioration status at the end of the associated trip. Utilization data is also used to update the forecasted utilization rates.

Periodic preventive maintenance, possibly with replacement, can update the current component status.

Inspection results (e.g., tire tread depth) can be compared with predicted values for that time and differences used to update the deterioration parameters of the equations.

Periodically all end-of-life calculations for the components of an asset are compared against the end of the following week, and if one of the EoL calculations occurs prior to the end of next week, a process will be invoked to prepare a work order for that asset covering all components that are near their EoL. If an EoL calculation is prior to the end of next week, an urgent Alert will be sent out to the maintenance supervisor.

The following diagram illustrates those processes for each tire on an asset:



Forecasted Utilization

Deterioration Prediction

Initialization of parameters

Seems Like Usage

Calculated End-of-Life (EoL)

Update Deterioration parameters

- Inspection vs. Prediction
- Resets after replacement/ repair

Alert generation