TEST LIBRERÍA PI-OSISOFT (AF Client) + PYTHON

creada por: Roberto Sánchez (rg.sanchez.a@gmail.com)

Es necesario para el uso de esta librería:

- Python 3.6 (por compatibilidad con Pythonnet)
- PI-SDK (version 2014 mínimo)
- AF-Client (version 2016 mínimo)
- Instalar Pythonnet: conda install -c pythonnet pythonnet

1. Importación de DLL y librerías

In [1]:
```
import pi_connect as p                       #Librería de conexión con Pi-Server
import pandas as pd                          #Librería de análisis de datos
import matplotlib.pyplot as plt              #Librería de gráficos (plots)
%matplotlib inline
```

2. Conexión servidor PI

In [2]:
```
pi_svr = p.PIserver()
```

3. Traer datos desde el servidor PI

In [3]:
```
tag_name = "CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV"            #Nombre de la Tag
time_range = pi_svr.time_range("2018-02-12", "2018-02-14")  #Rango de tiempo en el
que se desea consultar
pt = p.PI_point(pi_svr, tag_name)                          #Punto PI (None si no ex
iste)
```

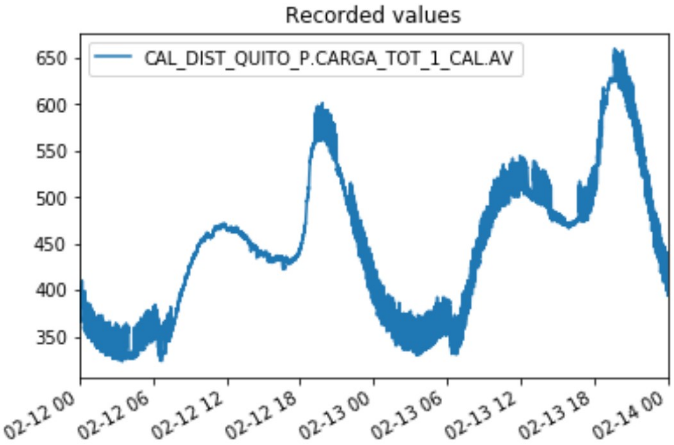
3.1. Valor en el momento

In [4]:
```
value1 = pt.snapshot()
print("value1:" + str(value1))
value2 = pt.current_value()
print("value2:" + str(value2))
```

```
value1:576.7722
value2:576.7722
```

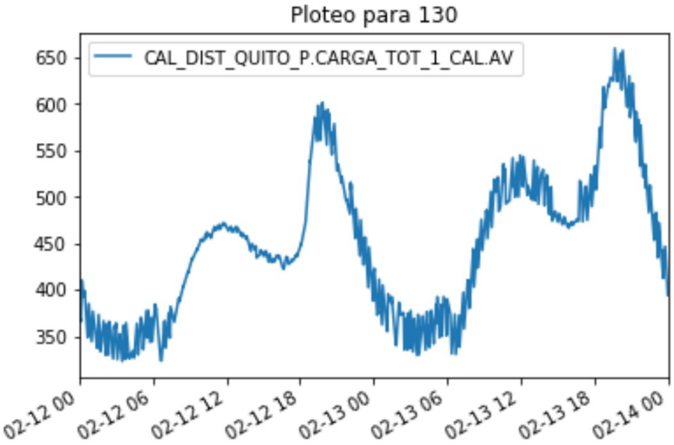
3.2. Valores guardados (registrados)

```
In [5]: df_raw = pt.recorded_values(time_range)
        df_raw.plot(title="Recorded values")

Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x17f4292bba8>
```



3.3. Diferentes maneras de traer datos:

```
In [6]: n_pixels = 130
        df_plot = pt.plot_values(time_range, n_pixels)
        df_plot.plot(title="Ploteo para " + str(n_pixels))

Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x17f43f6f0f0>
```



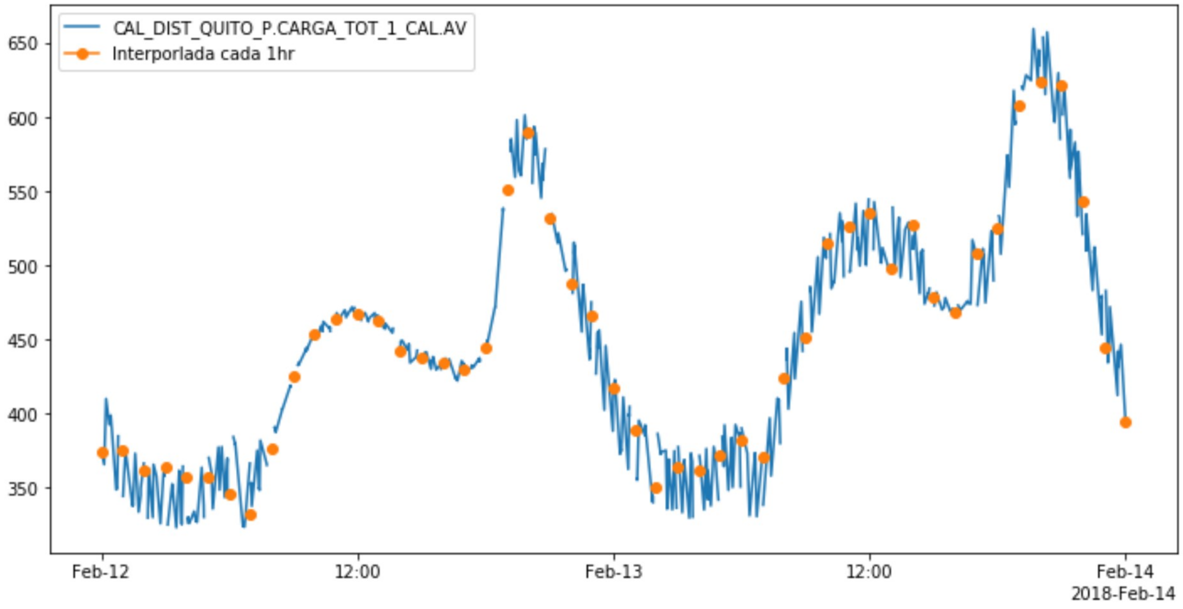
In [7]:

```
#Span en el que se desea realizar cálculos, interpolaciones, etc.
# df_plot = df_raw
span = pi_svr.span("1h")
# span = pi_svr.span("1h 30m")
df_interpolated = pt.interpolated(time_range, span)
int_label = "Interporlada cada " + str(span)
df_interpolated.columns = [int_label]
df_new = pd.concat([df_plot, df_interpolated], axis=1)
df_new.head(40)
```

Out[7]:

	CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV	Interporlada cada 1hr
2018-02-12 00:00:00	374.7252	374.7252
2018-02-12 00:07:56	365.7029	NaN
2018-02-12 00:13:16	410.0081	NaN
2018-02-12 00:22:04	393.3328	NaN
2018-02-12 00:22:16	392.5262	NaN
2018-02-12 00:26:36	398.8338	NaN
2018-02-12 00:41:44	348.7035	NaN
2018-02-12 00:44:16	349.3263	NaN
2018-02-12 00:44:24	349.4553	NaN
2018-02-12 00:47:16	384.8138	NaN
2018-02-12 01:00:00	NaN	375.8502
2018-02-12 01:00:56	344.3187	NaN
2018-02-12 01:06:24	369.3279	NaN
2018-02-12 01:06:36	369.2850	NaN
2018-02-12 01:07:16	376.5108	NaN
2018-02-12 01:27:24	337.8579	NaN
2018-02-12 01:28:36	339.5704	NaN
2018-02-12 01:28:44	337.7782	NaN
2018-02-12 01:35:04	373.1414	NaN
2018-02-12 01:44:44	333.6454	NaN
2018-02-12 01:50:44	344.9850	NaN
2018-02-12 01:50:56	349.8914	NaN
2018-02-12 01:59:24	366.4359	NaN
2018-02-12 02:00:00	NaN	361.7854
2018-02-12 02:10:36	329.6935	NaN
2018-02-12 02:12:44	359.5676	NaN
2018-02-12 02:12:56	362.8280	NaN
2018-02-12 02:24:36	329.9762	NaN
2018-02-12 02:26:36	365.6944	NaN
2018-02-12 02:35:04	357.7415	NaN
2018-02-12 02:35:16	356.8914	NaN
2018-02-12 02:46:04	325.8422	NaN
2018-02-12 02:54:44	360.0164	NaN
2018-02-12 02:57:04	361.8953	NaN
2018-02-12 02:57:16	357.9715	NaN
2018-02-12 02:59:56	364.2855	NaN
2018-02-12 03:00:00	NaN	364.2855
2018-02-12 03:06:44	325.0577	NaN
2018-02-12 03:19:12	352.5183	NaN
2018-02-12 03:19:24	352.0245	NaN

In [8]:
```
import matplotlib.dates as mdates
# seteo del tamaño de figura
plt.figure(figsize=(12, 6))
ax = plt.subplot(1, 1, 1)
ax.plot(df_new.index, df_new[tag_name])
ax.plot(df_new.index, df_new[int_label], marker='o')

# Creación de leyendas y personalización de ticks:
ax.legend([tag_name, int_label], loc='best')
# ticks periods = 4  determina 4 ticks
# lb = pd.date_range(df_new.index[0], df_new.index[-1], periods=5)
# lb = [x.strftime("%Y-%m-%d %H") for x in lb]
# ax.xticks(lb)

locator = mdates.AutoDateLocator(minticks=3, maxticks=7)
formatter = mdates.ConciseDateFormatter(locator)
ax.xaxis.set_major_locator(locator)
ax.xaxis.set_major_formatter(formatter)
```



In [9]:

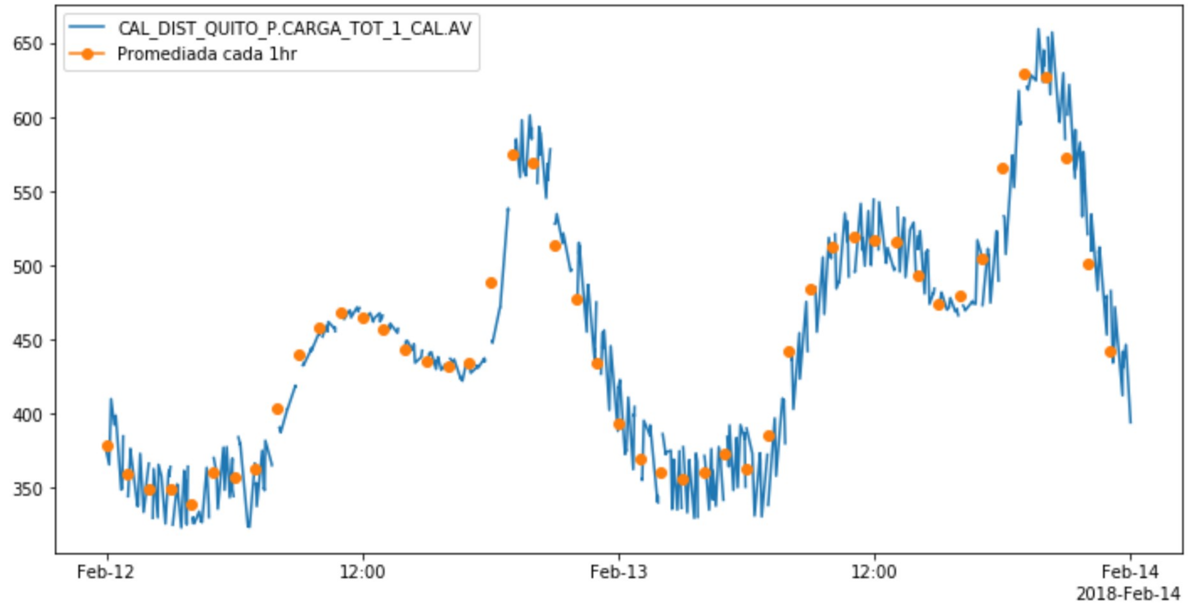
```
df_average = pt.average(time_range, span)
lb_av = "Promediada cada " + str(span)
df_average.columns = [lb_av]
df_new = pd.concat([df_plot, df_average], axis=1)
df_new.head(40)
```

Out[9]:

	CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV	Promediada cada 1hr
2018-02-12 00:00:00	374.7252	378.666455
2018-02-12 00:07:56	365.7029	NaN
2018-02-12 00:13:16	410.0081	NaN
2018-02-12 00:22:04	393.3328	NaN
2018-02-12 00:22:16	392.5262	NaN
2018-02-12 00:26:36	398.8338	NaN
2018-02-12 00:41:44	348.7035	NaN
2018-02-12 00:44:16	349.3263	NaN
2018-02-12 00:44:24	349.4553	NaN
2018-02-12 00:47:16	384.8138	NaN
2018-02-12 01:00:00	NaN	359.292500
2018-02-12 01:00:56	344.3187	NaN
2018-02-12 01:06:24	369.3279	NaN
2018-02-12 01:06:36	369.2850	NaN
2018-02-12 01:07:16	376.5108	NaN
2018-02-12 01:27:24	337.8579	NaN
2018-02-12 01:28:36	339.5704	NaN
2018-02-12 01:28:44	337.7782	NaN
2018-02-12 01:35:04	373.1414	NaN
2018-02-12 01:44:44	333.6454	NaN
2018-02-12 01:50:44	344.9850	NaN
2018-02-12 01:50:56	349.8914	NaN
2018-02-12 01:59:24	366.4359	NaN
2018-02-12 02:00:00	NaN	349.428504
2018-02-12 02:10:36	329.6935	NaN
2018-02-12 02:12:44	359.5676	NaN
2018-02-12 02:12:56	362.8280	NaN
2018-02-12 02:24:36	329.9762	NaN
2018-02-12 02:26:36	365.6944	NaN
2018-02-12 02:35:04	357.7415	NaN
2018-02-12 02:35:16	356.8914	NaN
2018-02-12 02:46:04	325.8422	NaN
2018-02-12 02:54:44	360.0164	NaN
2018-02-12 02:57:04	361.8953	NaN
2018-02-12 02:57:16	357.9715	NaN
2018-02-12 02:59:56	364.2855	NaN
2018-02-12 03:00:00	NaN	349.322400
2018-02-12 03:06:44	325.0577	NaN
2018-02-12 03:19:12	352.5183	NaN
2018-02-12 03:19:24	352.0245	NaN

In [10]:

```
plt.figure(figsize=(12, 6))
ax = plt.subplot(1, 1, 1)
ax.plot(df_new.index, df_new[tag_name])
ax.plot(df_new.index, df_new[lb_av], marker='o')

# Creación de leyendas y personalización de ticks:
ax.legend([tag_name, lb_av], loc='best')
locator = mdates.AutoDateLocator(minticks=3, maxticks=7)
formatter = mdates.ConciseDateFormatter(locator)
ax.xaxis.set_major_locator(locator)
ax.xaxis.set_major_formatter(formatter)
```



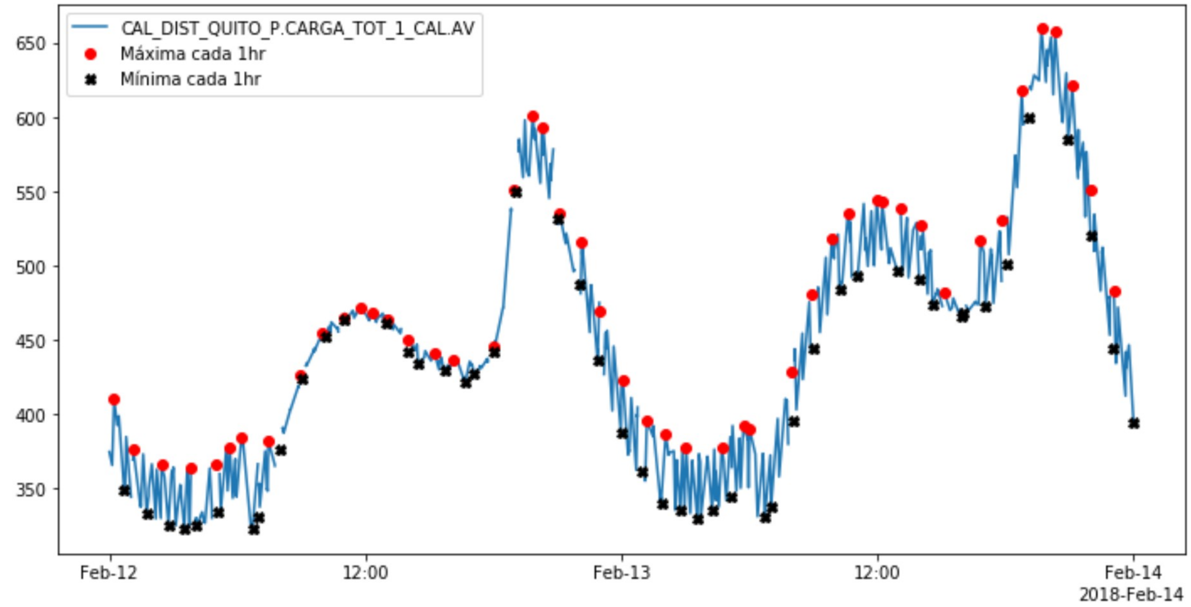
In [11]:

```
df_max = pt.max(time_range, span)
df_min = pt.min(time_range, span)
mx_lb = "Máxima cada " + str(span)
mn_lb = "Mínima cada " + str(span)
df_max.columns = [mx_lb]
df_min.columns = [mn_lb]
df_new = pd.concat([df_plot, df_max, df_min], axis=1)
df_new.head(40)
```

Out[11]:

	CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV	Máxima cada 1hr	Mínima cada 1hr
2018-02-12 00:00:00	374.7252	NaN	NaN
2018-02-12 00:07:56	365.7029	NaN	NaN
2018-02-12 00:13:16	410.0081	410.008057	NaN
2018-02-12 00:22:04	393.3328	NaN	NaN
2018-02-12 00:22:16	392.5262	NaN	NaN
2018-02-12 00:26:36	398.8338	NaN	NaN
2018-02-12 00:41:44	348.7035	NaN	348.703491
2018-02-12 00:44:16	349.3263	NaN	NaN
2018-02-12 00:44:24	349.4553	NaN	NaN
2018-02-12 00:47:16	384.8138	NaN	NaN
2018-02-12 01:00:56	344.3187	NaN	NaN
2018-02-12 01:05:36	NaN	376.986511	NaN
2018-02-12 01:06:24	369.3279	NaN	NaN
2018-02-12 01:06:36	369.2850	NaN	NaN
2018-02-12 01:07:16	376.5108	NaN	NaN
2018-02-12 01:27:24	337.8579	NaN	NaN
2018-02-12 01:28:36	339.5704	NaN	NaN
2018-02-12 01:28:44	337.7782	NaN	NaN
2018-02-12 01:35:04	373.1414	NaN	NaN
2018-02-12 01:44:44	333.6454	NaN	333.645447
2018-02-12 01:50:44	344.9850	NaN	NaN
2018-02-12 01:50:56	349.8914	NaN	NaN
2018-02-12 01:59:24	366.4359	NaN	NaN
2018-02-12 02:10:36	329.6935	NaN	NaN
2018-02-12 02:12:44	359.5676	NaN	NaN
2018-02-12 02:12:56	362.8280	NaN	NaN
2018-02-12 02:24:36	329.9762	NaN	NaN
2018-02-12 02:26:36	365.6944	365.694397	NaN
2018-02-12 02:35:04	357.7415	NaN	NaN
2018-02-12 02:35:16	356.8914	NaN	NaN
2018-02-12 02:46:04	325.8422	NaN	325.842163
2018-02-12 02:54:44	360.0164	NaN	NaN
2018-02-12 02:57:04	361.8953	NaN	NaN
2018-02-12 02:57:16	357.9715	NaN	NaN
2018-02-12 02:59:56	364.2855	NaN	NaN
2018-02-12 03:06:44	325.0577	NaN	NaN
2018-02-12 03:19:12	352.5183	NaN	NaN
2018-02-12 03:19:24	352.0245	NaN	NaN
2018-02-12 03:30:44	323.1518	NaN	323.151764
2018-02-12 03:38:36	361.6958	NaN	NaN

In [12]:
```
plt.figure(figsize=(12, 6))
ax = plt.subplot(1, 1, 1)
ax.plot(df_new.index, df_new[tag_name])
ax.plot(df_new.index, df_new[mx_lb], 'ro')
ax.plot(df_new.index, df_new[mn_lb], 'kX')


# Creación de leyendas y personalización de ticks:
ax.legend([tag_name, mx_lb, mn_lb], loc='best')
locator = mdates.AutoDateLocator(minticks=3, maxticks=7)
formatter = mdates.ConciseDateFormatter(locator)
ax.xaxis.set_major_locator(locator)
ax.xaxis.set_major_formatter(formatter)
```



3.4. Cálculos avanzados:

Cálculo del cumplimiento de una condición en un periodo de tiempo:

In [13]:

```
# limit = 324
limit = 350
filter_expression = "'{0}' < {1}".format(tag_name, limit)
print("La expresión a usar es: " + "\n\n" + filter_expression)
df_filter = pt.recorded_values(time_range, filterExpression=filter_expression)
lb_fil = filter_expression
df_filter.columns = [lb_fil]
df_filter
```

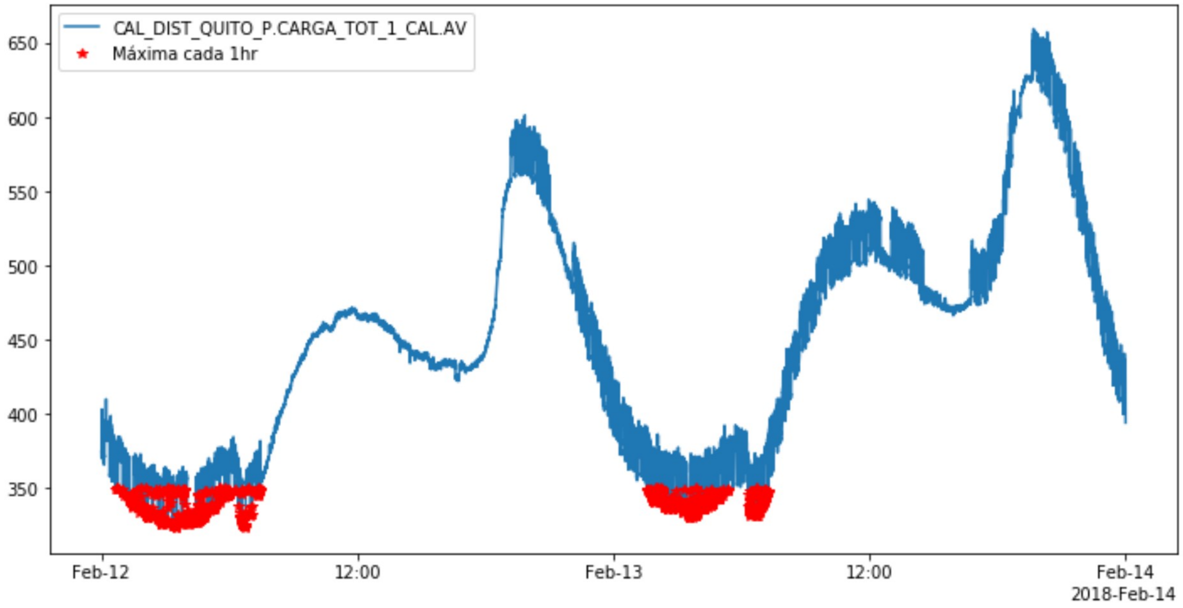
```
La expresión a usar es:

'CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV' < 350
```

Out[13]:

	'CAL_DIST_QUITO_P.CARGA_TOT_1_CAL.AV' < 350
2018-02-12 00:41:44	348.7035
2018-02-12 00:42:36	349.9134
2018-02-12 00:42:44	349.2024
2018-02-12 00:42:56	348.7139
2018-02-12 00:43:04	349.3289
...	...
2018-02-13 07:06:12	348.9105
2018-02-13 07:15:12	346.8138
2018-02-13 07:15:24	347.4648
2018-02-13 07:15:32	346.5060
2018-02-13 07:15:44	348.5477

1310 rows × 1 columns

In [14]:
```
df_new = pd.concat([df_raw, df_filter], axis=1)
n_point = 90000
df_new = df_new[:n_point]
plt.figure(figsize=(12, 6))
ax = plt.subplot(1, 1, 1)
ax.plot(df_new.index, df_new[tag_name])
ax.plot(df_new.index, df_new[lb_fil], 'r*')

# Creación de leyendas y personalización de ticks:
ax.legend([tag_name, mx_lb, mn_lb], loc='best')
locator = mdates.AutoDateLocator(minticks=3, maxticks=7)
formatter = mdates.ConciseDateFormatter(locator)
ax.xaxis.set_major_locator(locator)
ax.xaxis.set_major_formatter(formatter)
```

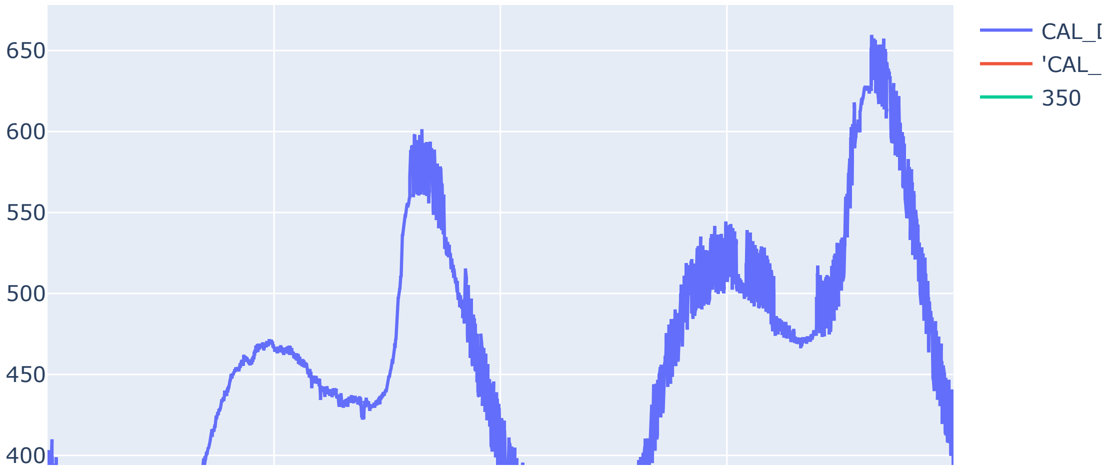


In [15]:
```
# instalar plolty si no está disponible:
import subprocess as sub
sub.run("pip install plotly")
```

Out[15]:
```
CompletedProcess(args='pip install plotly', returncode=0)
```

In [16]:

```
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
plot1 = go.Scatter(x=df_new.index, y=df_new[tag_name], name=tag_name)
plot2 = go.Scatter(x=df_new.index, y=df_new[lb_fil], name=lb_fil)
plot3 = go.Scatter(x=df_new.index, y=[limit]*len(df_new.index), name=str(limit))
iplot([plot1, plot2, plot3])
```



In [17]:

```
plot([plot1, plot2, plot3])
```

Out[17]:

```
'temp-plot.html'
```

In [ ]:

```

```