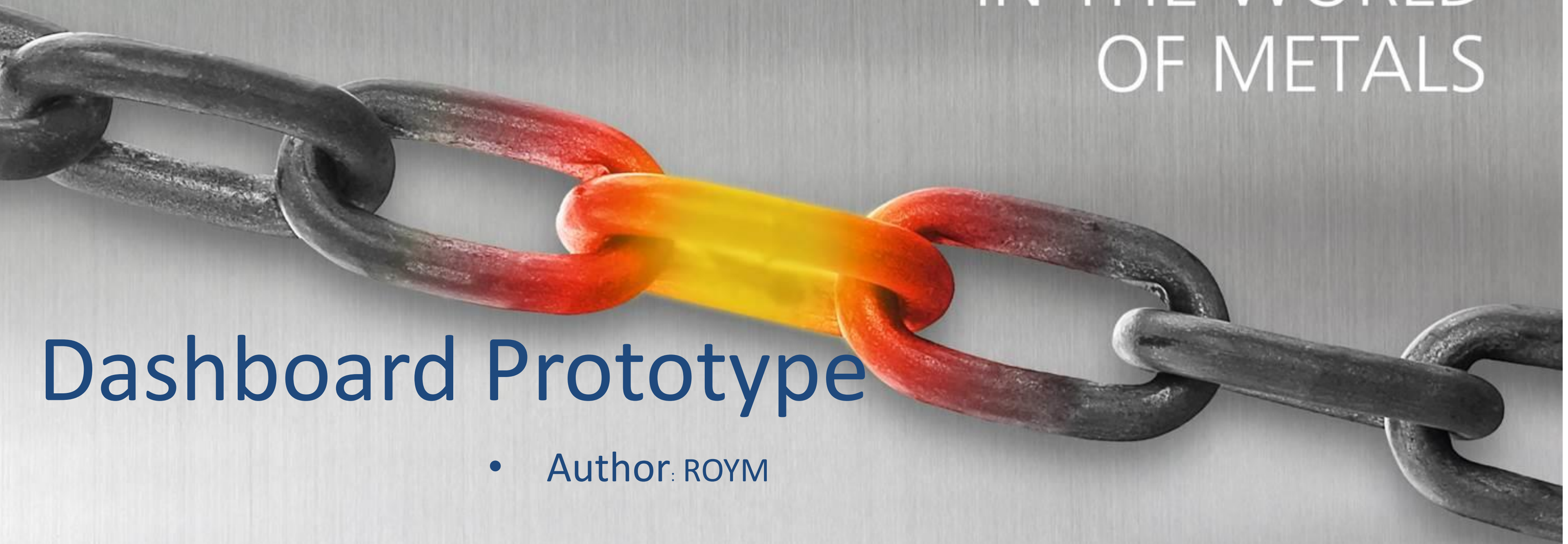
LEADING PARTNER
IN THE WORLD
OF METALS



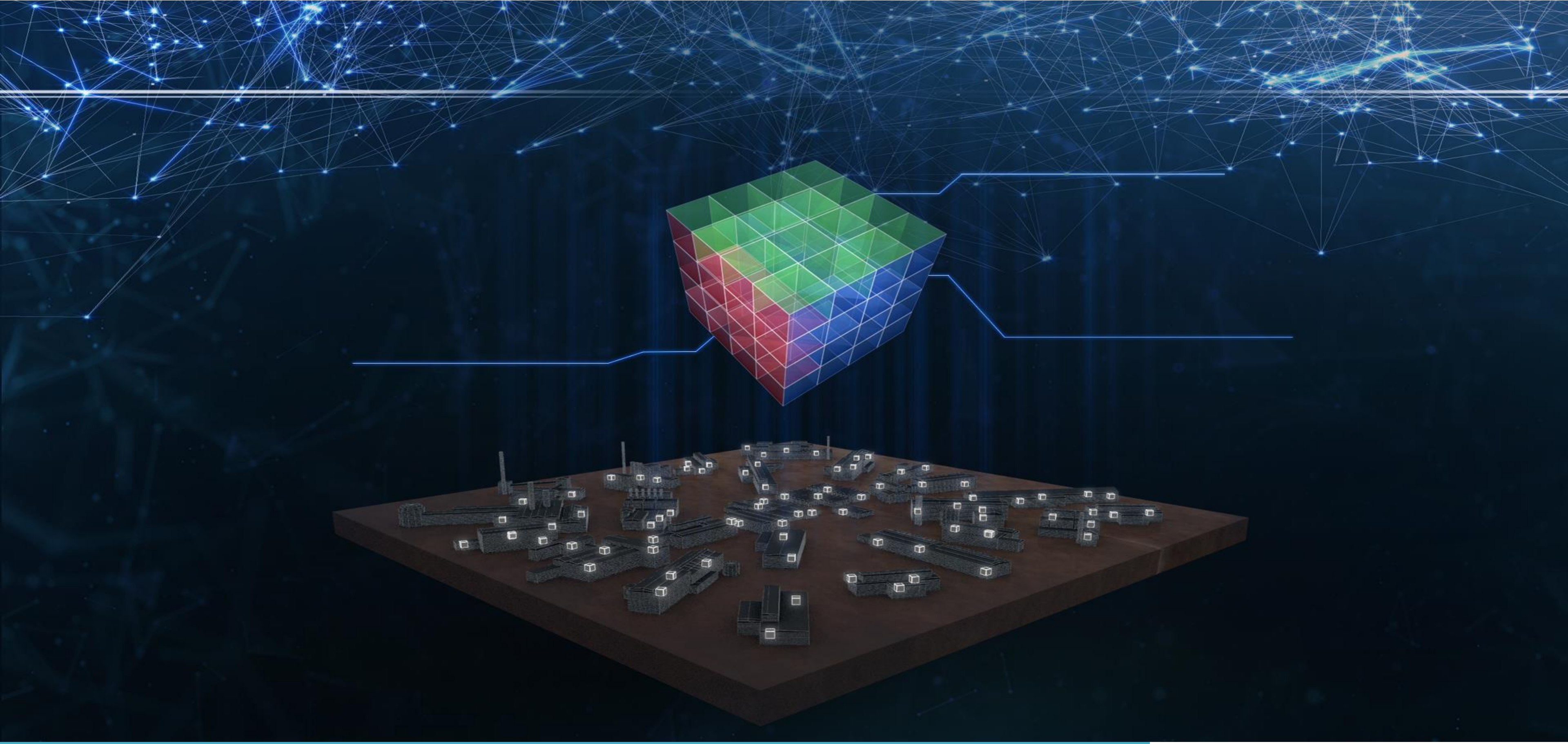
Dashboard Prototype

- Author: ROYM

SMS group

- Confidential

Digitalization makes it possible to take different points of view



Dashboard’s Powered by Plotly, Python & Panda

Introduction

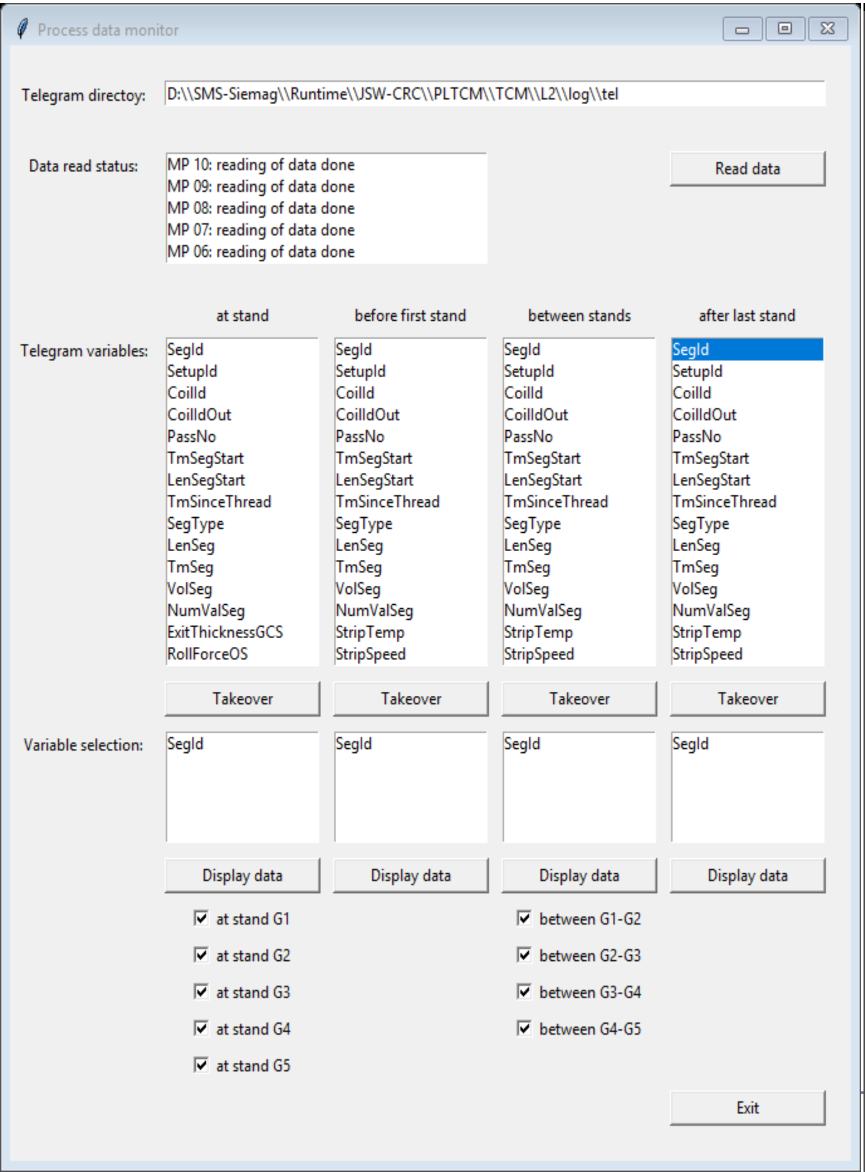
While working on PLTCM Web HMI project I came across data visualization tool used in this project. The tool simply uses the matplotlib(open source lib for plotting graphs) to plot data from data's received from respective machine.

However this tool require full user intervention to load the data from a flat file system. As data's are huge it's take ample amount of time before rendering the plot.

Bottleneck

It's good tool to visualize the data from machine and verify the various process event that has happened during the process. But at same time it's not automated, Interactive and real time as user has to refresh the data every time by loading it to form.

Old visualization tool was based on python and Matplotlib



Telegram directoy: D:\\SMS-Siemag\\Runtime\\JSW-CRC\\PLTCM\\TCM\\L2\\log\\tel

Data read status:
MP 10: reading of data done
MP 09: reading of data done
MP 08: reading of data done
MP 07: reading of data done
MP 06: reading of data done

Telegram variables:

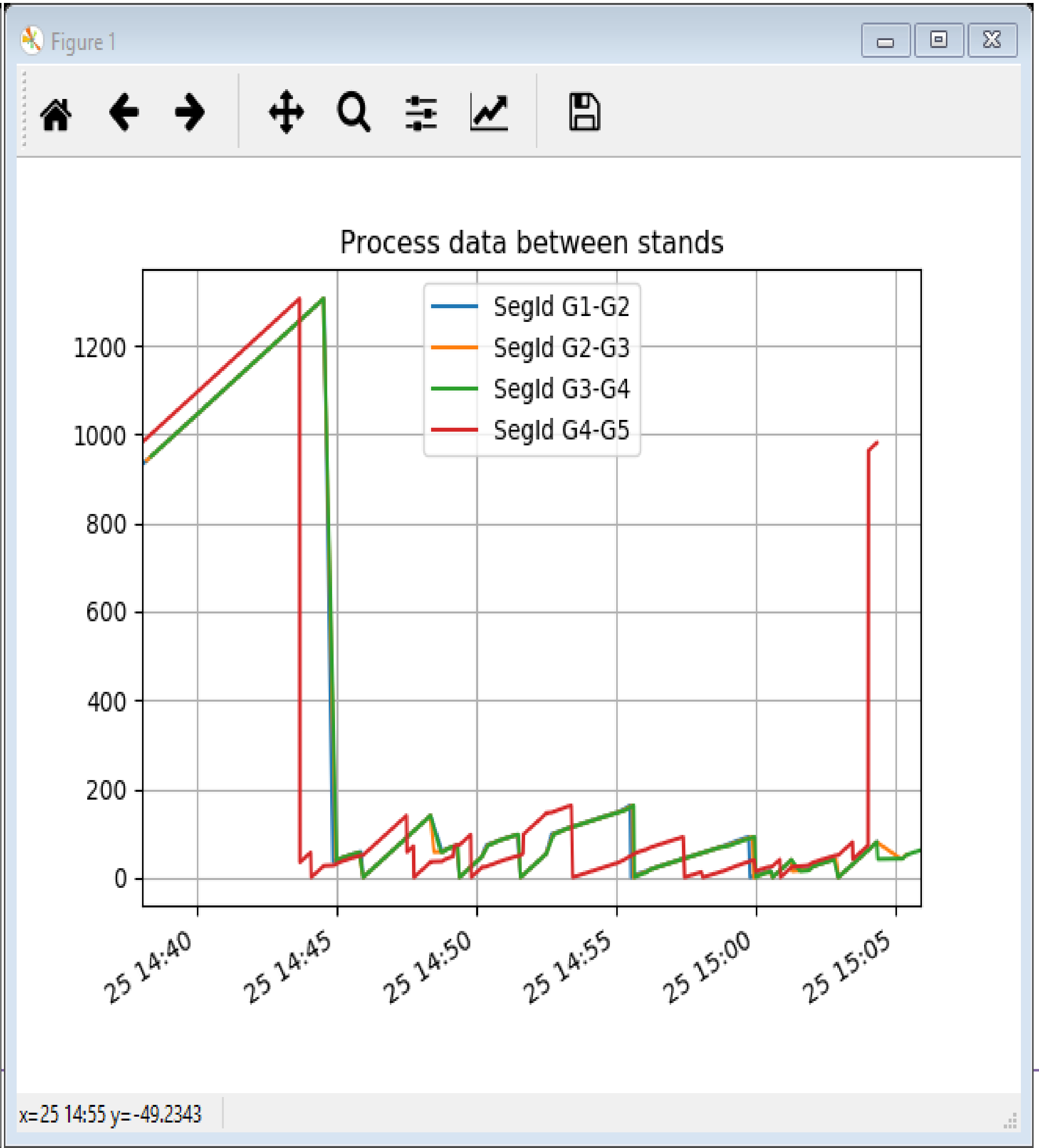
at stand	before first stand	between stands	after last stand
SegId	SegId	SegId	SegId
SetupId	SetupId	SetupId	SetupId
CoilId	CoilId	CoilId	CoilId
CoilIdOut	CoilIdOut	CoilIdOut	CoilIdOut
PassNo	PassNo	PassNo	PassNo
TmSegStart	TmSegStart	TmSegStart	TmSegStart
LenSegStart	LenSegStart	LenSegStart	LenSegStart
TmSinceThread	TmSinceThread	TmSinceThread	TmSinceThread
SegType	SegType	SegType	SegType
LenSeg	LenSeg	LenSeg	LenSeg
TmSeg	TmSeg	TmSeg	TmSeg
VolSeg	VolSeg	VolSeg	VolSeg
NumValSeg	NumValSeg	NumValSeg	NumValSeg
ExitThicknessGCS	StripTemp	StripTemp	StripTemp
RollForceOS	StripSpeed	StripSpeed	StripSpeed

Variable selection:

SegId	SegId	SegId	SegId

☑ at stand G1
☑ at stand G2
☑ at stand G3
☑ at stand G4
☑ at stand G5

☑ between G1-G2
☑ between G2-G3
☑ between G3-G4
☑ between G4-G5



x=25 14:55 y=-49.2343

Enhancement

Plotly | Dash: - Dash is a Python framework for building analytical web applications. No JavaScript required build on top of Plotly.js, React, and Flask, Dash ties modern UI elements like dropdowns, sliders, and graphs directly to your analytical python code.

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python.

pandas is a Python package License under BSD 3-Clause.
http://pandas.pydata.org/pandas-docs/stable/getting_started/overview.html#license

Plotly | Dash is License under MIT.
https://github.com/plotly/dash/blob/master/LICENSE

NumPy: -NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

Celery: Distributed Task Queue
Celery is an asynchronous task queue/job queue based on distributed message passing. It is focused on real-time operation, but supports scheduling as well.
The execution units, called tasks, are executed concurrently on a single or more worker servers using multiprocessing, Eventlet, or gevent. Tasks can execute asynchronously (in the background) or synchronously (wait until ready).

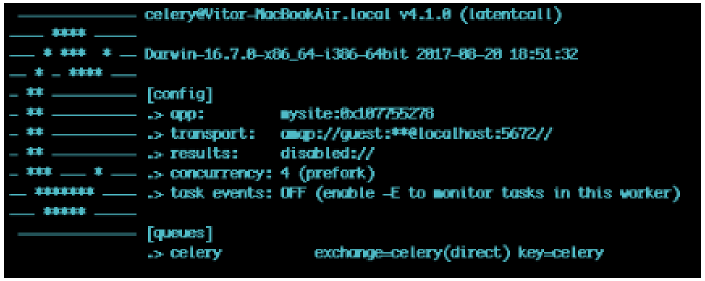
Celery is licensed under The BSD License (3 Clause).
https://github.com/celery/celery/blob/master/LICENSE

NumPy is licensed under the BSD license, enabling reuse with few restrictions.
https://www.numpy.org/license.html

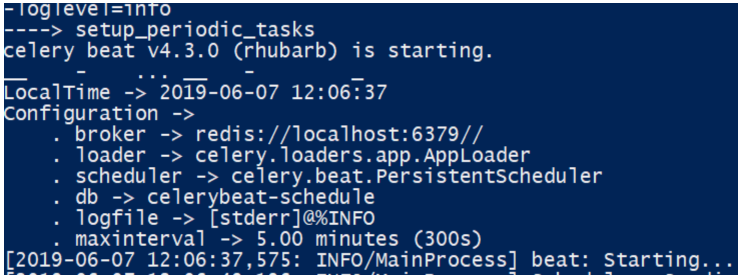
Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes with radius queries and streams. Redis has built-in replication, Lua scripting, LRU eviction, transactions and different levels of on-disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster.

https://redis.io/topics/license

Application Architecture



```
 -------------- celery@Vitor-MacBookAir.local v4.1.0 (latentcall)
---- **** -----
--- * ***  * -- Darwin-16.7.0-x86_64-i386-64bit 2017-08-20 18:51:32
-- * - **** ---
- ** ---------- [config]
- ** ---------- .> app:         mysite:0x107755278
- ** ---------- .> transport:   amqp://guest:**@localhost:5672//
- ** ---------- .> results:     disabled://
- *** --- * --- .> concurrency: 4 (prefork)
-- ******* ---- .> task events: OFF (enable -E to monitor tasks in this worker)
--- ***** -----
 -------------- [queues]
                .> celery           exchange=celery(direct) key=celery
```



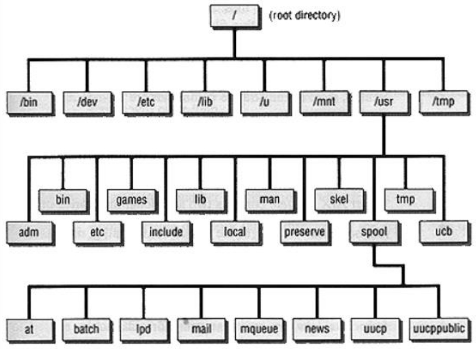
```
-loglevel=info
----> setup_periodic_tasks
celery beat v4.3.0 (rhubarb) is starting.
__    -    ... __   -        _
LocalTime -> 2019-06-07 12:06:37
Configuration ->
    . broker -> redis://localhost:6379//
    . loader -> celery.loaders.app.AppLoader
    . scheduler -> celery.beat.PersistentScheduler
    . db -> celerybeat-schedule
    . logfile -> [stderr]@%INFO
    . maxinterval -> 5.00 minutes (300s)
[2019-06-07 12:06:37,575: INFO/MainProcess] beat: Starting...
```

Celery Worker

Celery Beat

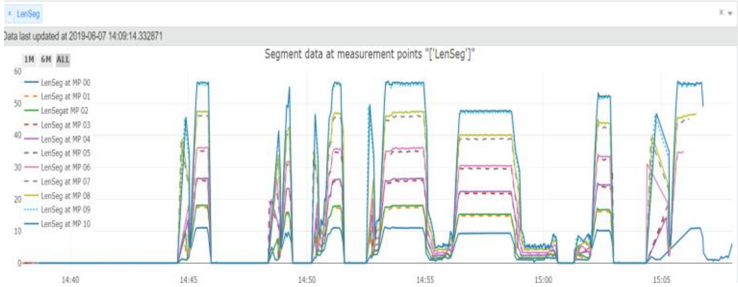
```
Redis 3.2.8 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 1

      http://redis.io
```

File System

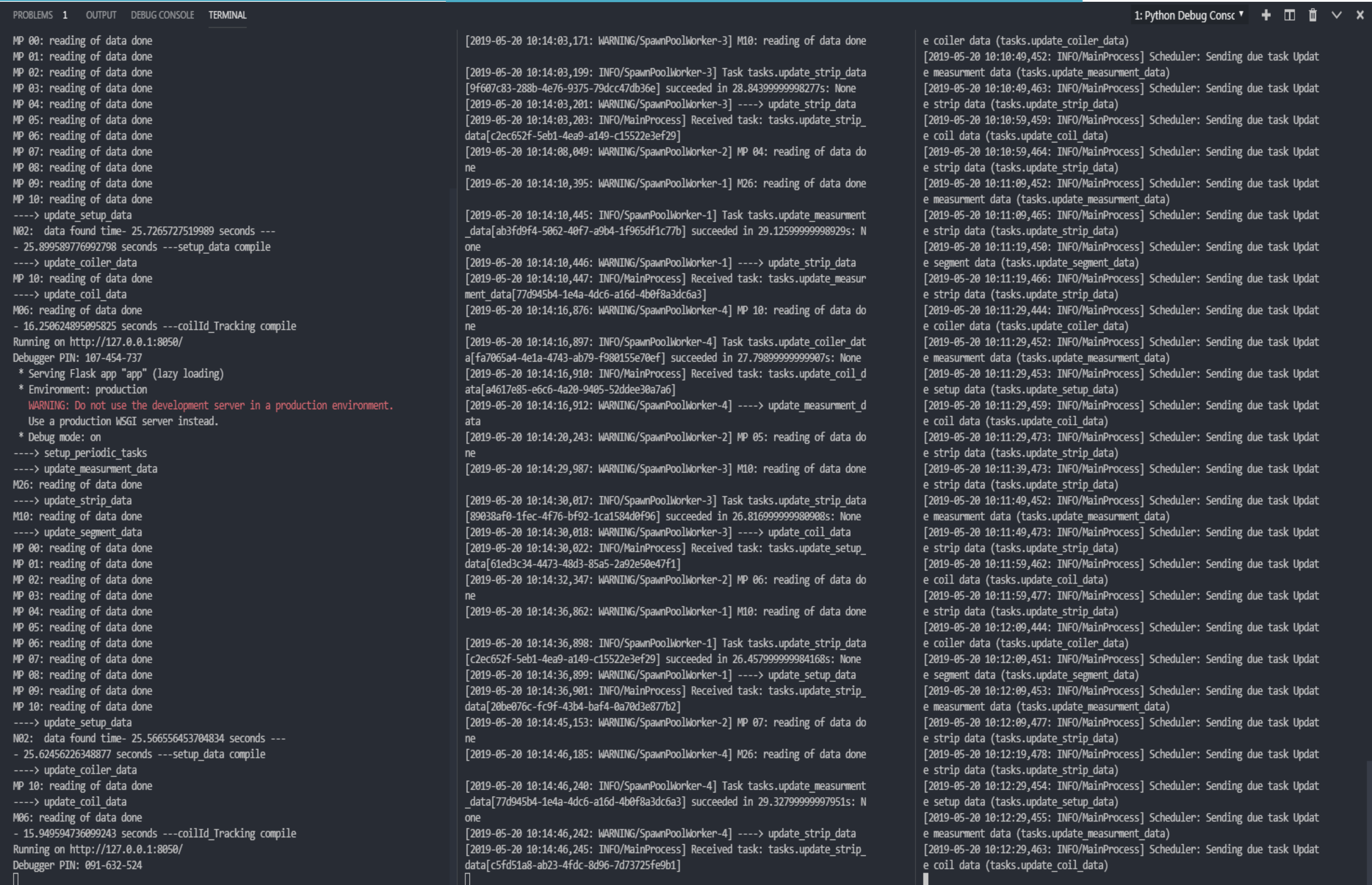


Redis Database

Dash App



Data last updated at 2019-06-07 14:09:14.332871



Flask Application

```
MP 00: reading of data done
MP 01: reading of data done
MP 02: reading of data done
MP 03: reading of data done
MP 04: reading of data done
MP 05: reading of data done
MP 06: reading of data done
MP 07: reading of data done
MP 08: reading of data done
MP 09: reading of data done
MP 10: reading of data done
----> update_setup_data
N02:  data found time- 25.7265727519989 seconds ---
- 25.899589776992798 seconds ---setup_data compile
----> update_coiler_data
MP 10: reading of data done
----> update_coil_data
M06: reading of data done
- 16.250624895095825 seconds ---coilId_Tracking compile
Running on http://127.0.0.1:8050/
Debugger PIN: 107-454-737
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
----> setup_periodic_tasks
----> update_measurment_data
M26: reading of data done
----> update_strip_data
M10: reading of data done
----> update_segment_data
MP 00: reading of data done
MP 01: reading of data done
MP 02: reading of data done
MP 03: reading of data done
MP 04: reading of data done
MP 05: reading of data done
MP 06: reading of data done
MP 07: reading of data done
MP 08: reading of data done
MP 09: reading of data done
MP 10: reading of data done
----> update_setup_data
N02:  data found time- 25.566556453704834 seconds ---
- 25.62456226348877 seconds ---setup_data compile
----> update_coiler_data
MP 10: reading of data done
----> update_coil_data
M06: reading of data done
- 15.949594736099243 seconds ---coilId_Tracking compile
Running on http://127.0.0.1:8050/
Debugger PIN: 091-632-524
```

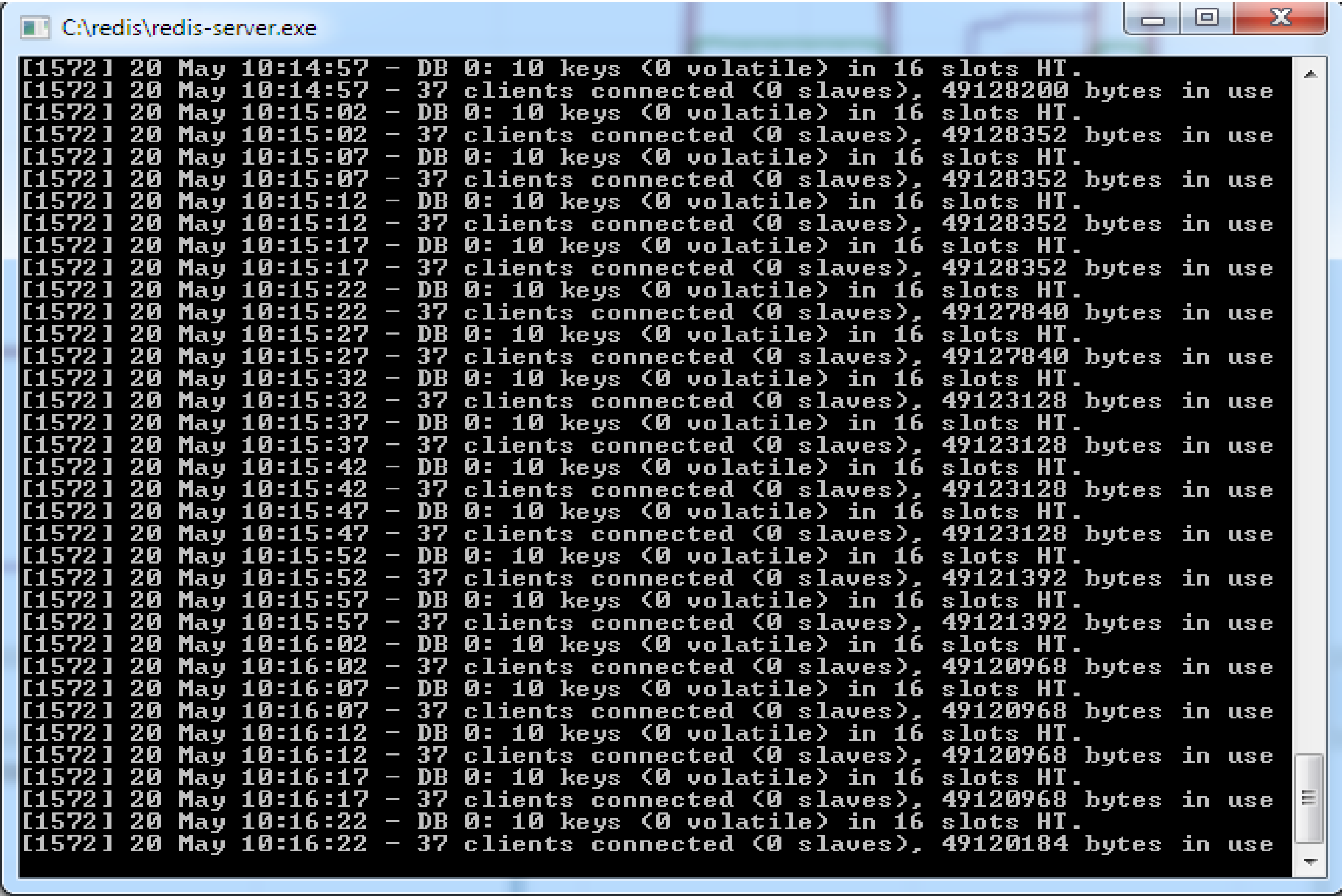
Celery Worker

```
[2019-05-20 10:14:03,171: WARNING/SpawnPoolWorker-3] M10: reading of data done

[2019-05-20 10:14:03,199: INFO/SpawnPoolWorker-3] Task tasks.update_strip_data
[9f607c83-288b-4e76-9375-79dcc47db36e] succeeded in 28.84399999998277s: None
[2019-05-20 10:14:03,201: WARNING/SpawnPoolWorker-3] ----> update_strip_data
[2019-05-20 10:14:03,203: INFO/MainProcess] Received task: tasks.update_strip_
data[c2ec652f-5eb1-4ea9-a149-c15522e3ef29]
[2019-05-20 10:14:08,049: WARNING/SpawnPoolWorker-2] MP 04: reading of data do
ne
[2019-05-20 10:14:10,395: WARNING/SpawnPoolWorker-1] M26: reading of data done

[2019-05-20 10:14:10,445: INFO/SpawnPoolWorker-1] Task tasks.update_measurment
_data[ab3fd9f4-5062-40f7-a9b4-1f965df1c77b] succeeded in 29.12599999998929s: N
one
[2019-05-20 10:14:10,446: WARNING/SpawnPoolWorker-1] ----> update_strip_data
[2019-05-20 10:14:10,447: INFO/MainProcess] Received task: tasks.update_measur
ment_data[77d945b4-1e4a-4dc6-a16d-4b0f8a3dc6a3]
[2019-05-20 10:14:16,876: WARNING/SpawnPoolWorker-4] MP 10: reading of data do
ne
[2019-05-20 10:14:16,897: INFO/SpawnPoolWorker-4] Task tasks.update_coiler_dat
a[fa7065a4-4e1a-4743-ab79-f980155e70ef] succeeded in 27.79899999999907s: None
[2019-05-20 10:14:16,910: INFO/MainProcess] Received task: tasks.update_coil_d
ata[a4617e85-e6c6-4a20-9405-52ddee30a7a6]
[2019-05-20 10:14:16,912: WARNING/SpawnPoolWorker-4] ----> update_measurment_d
ata
[2019-05-20 10:14:20,243: WARNING/SpawnPoolWorker-2] MP 05: reading of data do
ne
[2019-05-20 10:14:29,987: WARNING/SpawnPoolWorker-3] M10: reading of data done

[2019-05-20 10:14:30,017: INFO/SpawnPoolWorker-3] Task tasks.update_strip_data
[89038af0-1fec-4f76-bf92-1ca1584d0f96] succeeded in 26.816999999980908s: None
[2019-05-20 10:14:30,018: WARNING/SpawnPoolWorker-3] ----> update_coil_data
[2019-05-20 10:14:30,022: INFO/MainProcess] Received task: tasks.update_setup_
data[61ed3c34-4473-48d3-85a5-2a92e50e47f1]
[2019-05-20 10:14:32,347: WARNING/SpawnPoolWorker-2] MP 06: reading of data do
ne
[2019-05-20 10:14:36,862: WARNING/SpawnPoolWorker-1] M10: reading of data done

[2019-05-20 10:14:36,898: INFO/SpawnPoolWorker-1] Task tasks.update_strip_data
[c2ec652f-5eb1-4ea9-a149-c15522e3ef29] succeeded in 26.457999999984168s: None
[2019-05-20 10:14:36,899: WARNING/SpawnPoolWorker-1] ----> update_setup_data
[2019-05-20 10:14:36,901: INFO/MainProcess] Received task: tasks.update_strip_
data[20be076c-fc9f-43b4-baf4-0a70d3e877b2]
[2019-05-20 10:14:45,153: WARNING/SpawnPoolWorker-2] MP 07: reading of data do
ne
[2019-05-20 10:14:46,185: WARNING/SpawnPoolWorker-4] M26: reading of data done

[2019-05-20 10:14:46,240: INFO/SpawnPoolWorker-4] Task tasks.update_measurment
_data[77d945b4-1e4a-4dc6-a16d-4b0f8a3dc6a3] succeeded in 29.32799999997951s: N
one
[2019-05-20 10:14:46,242: WARNING/SpawnPoolWorker-4] ----> update_strip_data
[2019-05-20 10:14:46,245: INFO/MainProcess] Received task: tasks.update_strip_
data[c5fd51a8-ab23-4fdc-8d96-7d73725fe9b1]
```

Celery Beat

```
e coiler data (tasks.update_coiler_data)
[2019-05-20 10:10:49,452: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:10:49,463: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:10:59,459: INFO/MainProcess] Scheduler: Sending due task Updat
e coil data (tasks.update_coil_data)
[2019-05-20 10:10:59,464: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:09,452: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:11:09,465: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:19,450: INFO/MainProcess] Scheduler: Sending due task Updat
e segment data (tasks.update_segment_data)
[2019-05-20 10:11:19,466: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:29,444: INFO/MainProcess] Scheduler: Sending due task Updat
e coiler data (tasks.update_coiler_data)
[2019-05-20 10:11:29,452: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:11:29,453: INFO/MainProcess] Scheduler: Sending due task Updat
e setup data (tasks.update_setup_data)
[2019-05-20 10:11:29,459: INFO/MainProcess] Scheduler: Sending due task Updat
e coil data (tasks.update_coil_data)
[2019-05-20 10:11:29,473: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:39,473: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:49,452: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:11:49,473: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:11:59,462: INFO/MainProcess] Scheduler: Sending due task Updat
e coil data (tasks.update_coil_data)
[2019-05-20 10:11:59,477: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:12:09,444: INFO/MainProcess] Scheduler: Sending due task Updat
e coiler data (tasks.update_coiler_data)
[2019-05-20 10:12:09,451: INFO/MainProcess] Scheduler: Sending due task Updat
e segment data (tasks.update_segment_data)
[2019-05-20 10:12:09,453: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:12:09,477: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:12:19,478: INFO/MainProcess] Scheduler: Sending due task Updat
e strip data (tasks.update_strip_data)
[2019-05-20 10:12:29,454: INFO/MainProcess] Scheduler: Sending due task Updat
e setup data (tasks.update_setup_data)
[2019-05-20 10:12:29,455: INFO/MainProcess] Scheduler: Sending due task Updat
e measurment data (tasks.update_measurment_data)
[2019-05-20 10:12:29,463: INFO/MainProcess] Scheduler: Sending due task Updat
e coil data (tasks.update_coil_data)
```

Redis Database



```
[1572] 20 May 10:14:57 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:14:57 - 37 clients connected (0 slaves), 49128200 bytes in use
[1572] 20 May 10:15:02 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:02 - 37 clients connected (0 slaves), 49128352 bytes in use
[1572] 20 May 10:15:07 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:07 - 37 clients connected (0 slaves), 49128352 bytes in use
[1572] 20 May 10:15:12 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:12 - 37 clients connected (0 slaves), 49128352 bytes in use
[1572] 20 May 10:15:17 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:17 - 37 clients connected (0 slaves), 49128352 bytes in use
[1572] 20 May 10:15:22 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:22 - 37 clients connected (0 slaves), 49127840 bytes in use
[1572] 20 May 10:15:27 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:27 - 37 clients connected (0 slaves), 49127840 bytes in use
[1572] 20 May 10:15:32 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:32 - 37 clients connected (0 slaves), 49123128 bytes in use
[1572] 20 May 10:15:37 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:37 - 37 clients connected (0 slaves), 49123128 bytes in use
[1572] 20 May 10:15:42 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:42 - 37 clients connected (0 slaves), 49123128 bytes in use
[1572] 20 May 10:15:47 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:47 - 37 clients connected (0 slaves), 49123128 bytes in use
[1572] 20 May 10:15:52 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:52 - 37 clients connected (0 slaves), 49121392 bytes in use
[1572] 20 May 10:15:57 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:15:57 - 37 clients connected (0 slaves), 49121392 bytes in use
[1572] 20 May 10:16:02 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:16:02 - 37 clients connected (0 slaves), 49120968 bytes in use
[1572] 20 May 10:16:07 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:16:07 - 37 clients connected (0 slaves), 49120968 bytes in use
[1572] 20 May 10:16:12 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:16:12 - 37 clients connected (0 slaves), 49120968 bytes in use
[1572] 20 May 10:16:17 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:16:17 - 37 clients connected (0 slaves), 49120968 bytes in use
[1572] 20 May 10:16:22 - DB 0: 10 keys (0 volatile) in 16 slots HT.
[1572] 20 May 10:16:22 - 37 clients connected (0 slaves), 49120184 bytes in use
```

Software Cost Matrix

Name	Description	License Type	Cost
Dash	Dash is a Python framework for building analytical web applications	Open Source	Nil
Plotly	online data analytics and visualization tools	Open Source	Nil
Pandas	**real world** data analysis in Python.	Open Source	Nil
Numpy	multi-dimensional Array Handling in Python	Open Source	Nil
Celery	Celery is an asynchronous task queue/job queue based on distributed message passing	Open Source	Nil
Redis	In-memory data structure store, used as a database, cache and message broker	Open Source	Nil

World of Metals

Dash will easily fit into our company requirement for data analytics and Production/Maintenance dashboard.

This could be our first data driven package for the steel plant where we can present the customer with the dashboard, Data visualization, Machine learning model output.

Dash App Architecture

Structuring a Multi-Page App

Here's how to structure a multi-page app, where each app is contained in a separate file.

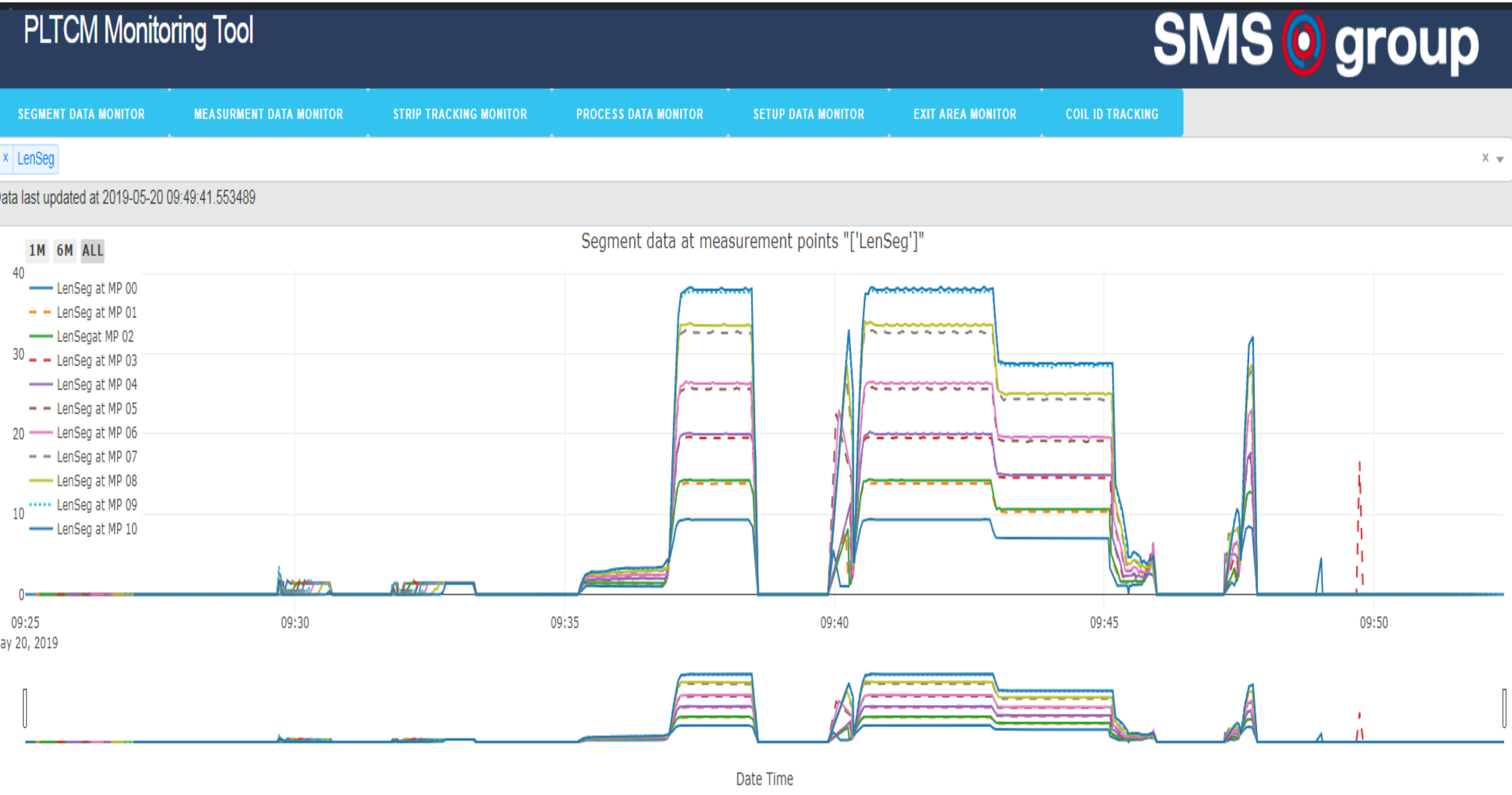
File structure:

```
-   app.py
-   index.py
-   apps
                -- __init__.py
                -- app1.py
                -- app2.py
```

It is worth noting that in both of these project structures, the Dash instance is defined in a separate app.py, while the entry point for running the app is index.py.

This separation is required to avoid circular imports: the files containing the callback definitions require access to the Dash app instance however if this were imported from index.py, the initial loading of index.py would ultimately require itself to be already imported, which cannot be satisfied.

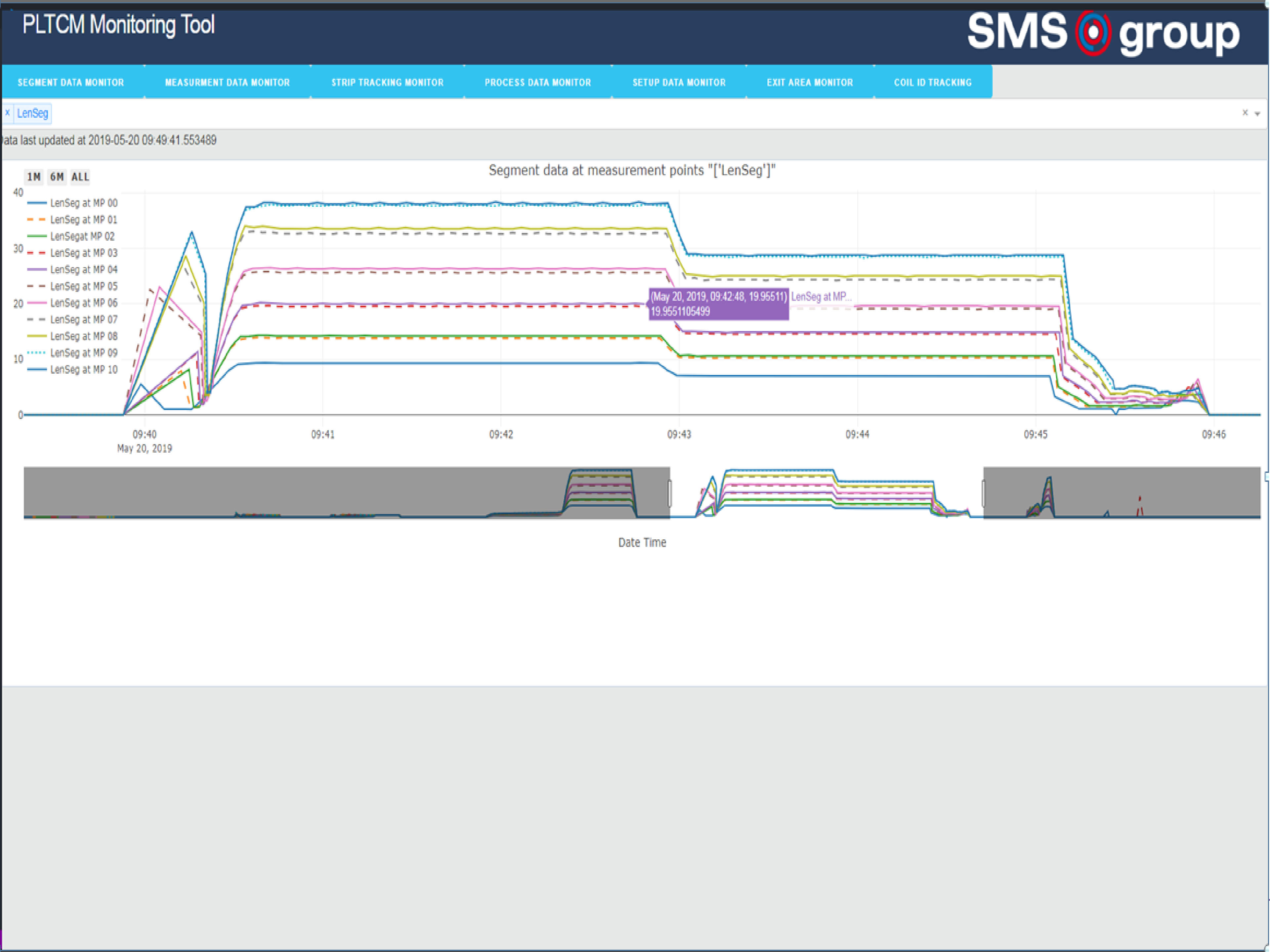
Machine Data Visualization



PLTCM Monitoring Tool

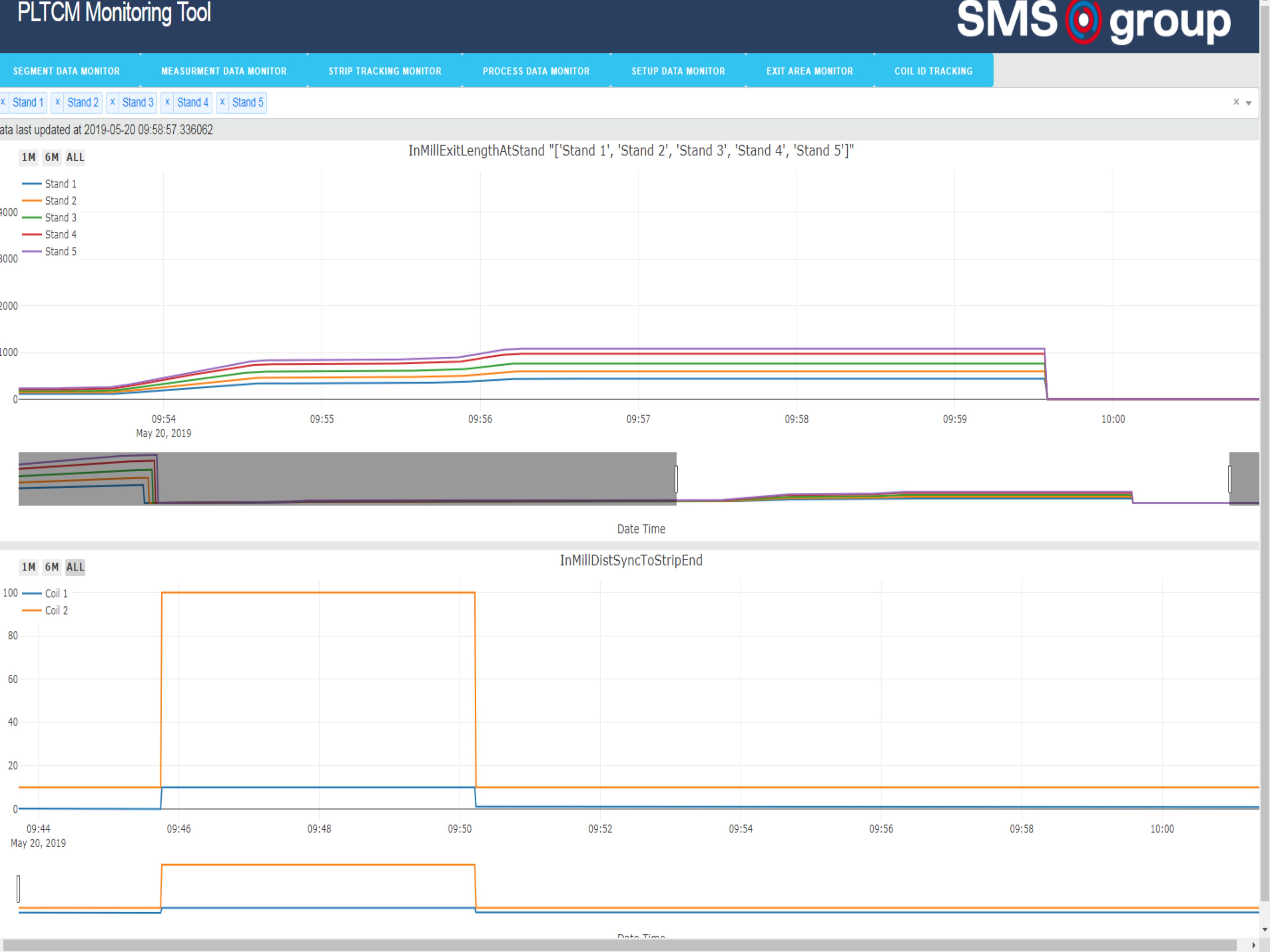
ata last updated at 2019-05-20 09:49:41.553489

Segment data at measurement points "['LenSeg']"



ata last updated at 2019-05-20 09:49:41.553489

Segment data at measurement points "['LenSeg']"

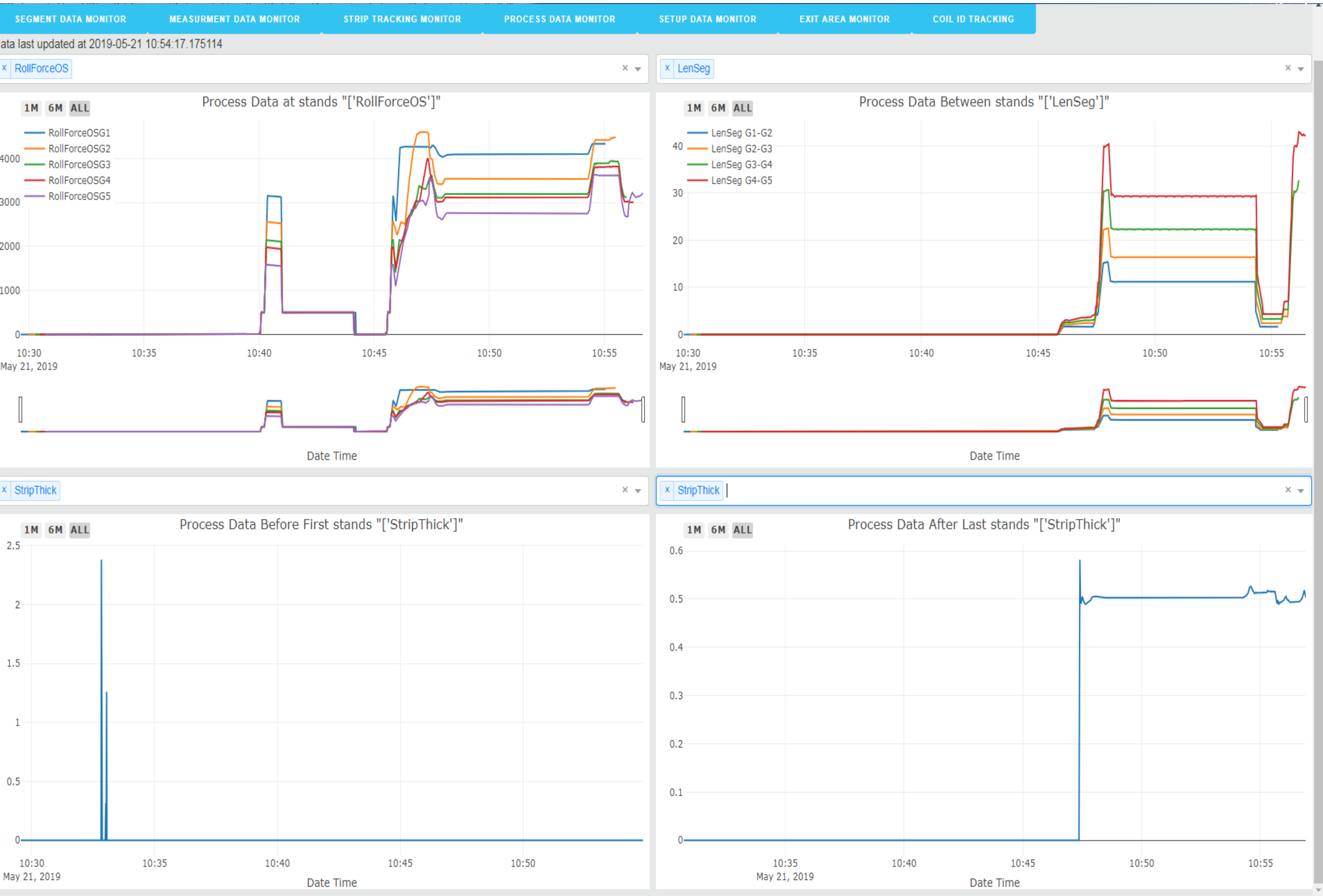


ata last updated at 2019-05-20 09:58:57.336062

InMillExitLengthAtStand "['Stand 1', 'Stand 2', 'Stand 3', 'Stand 4', 'Stand 5']"

InMillDistSyncToStripEnd

SMS group



SEGMENT DATA MONITOR | MEASURMENT DATA MONITOR | STRIP TRACKING MONITOR | PROCESS DATA MONITOR | SETUP DATA MONITOR | EXIT AREA MONITOR | COIL ID TRACKING

ata last updated at 2019-05-21 10:54:17.175114

× RollForceOS

× LenSeg

Process Data at stands "['RollForceOS']"

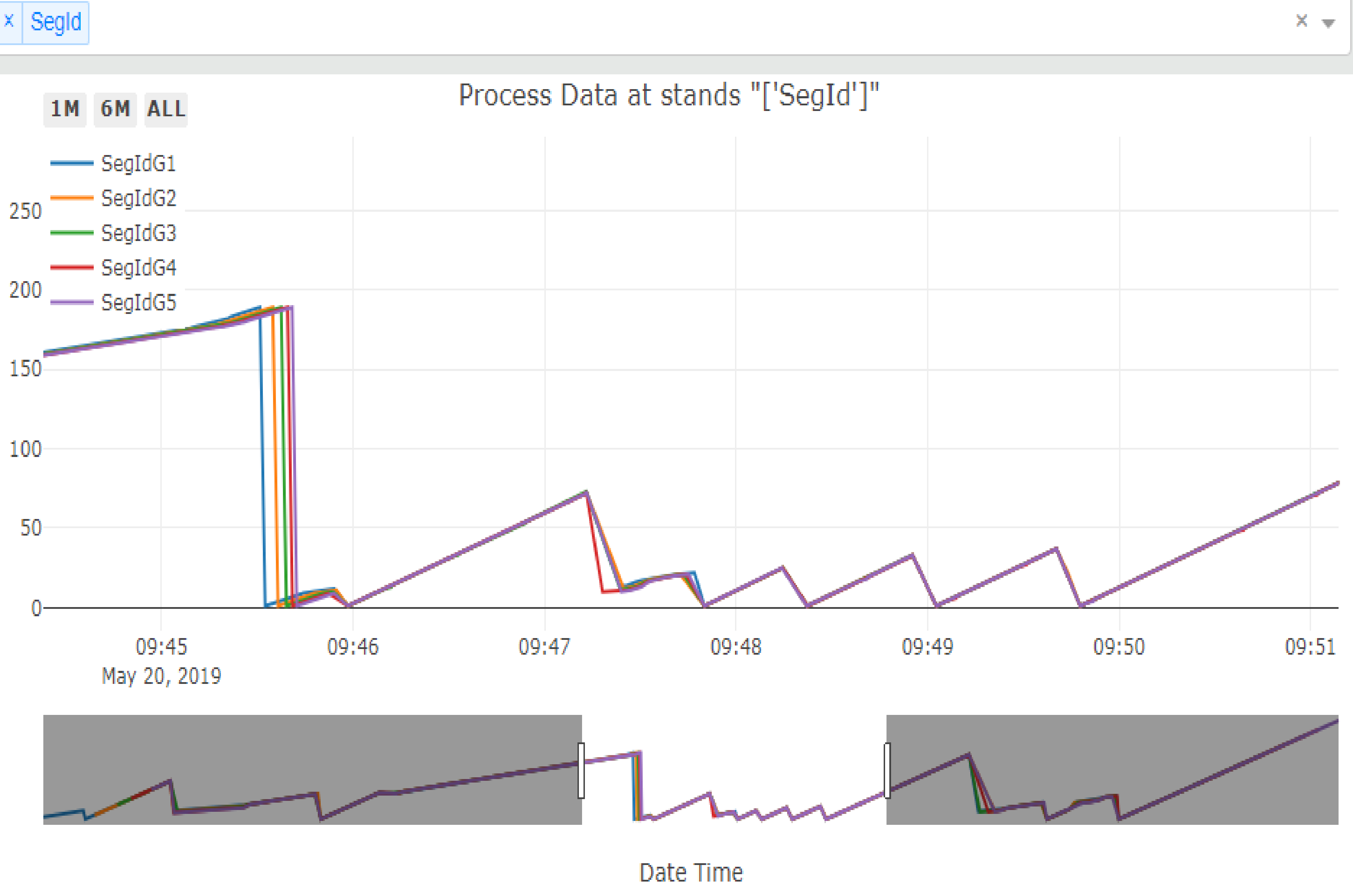
Process Data Between stands "['LenSeg']"

× StripThick

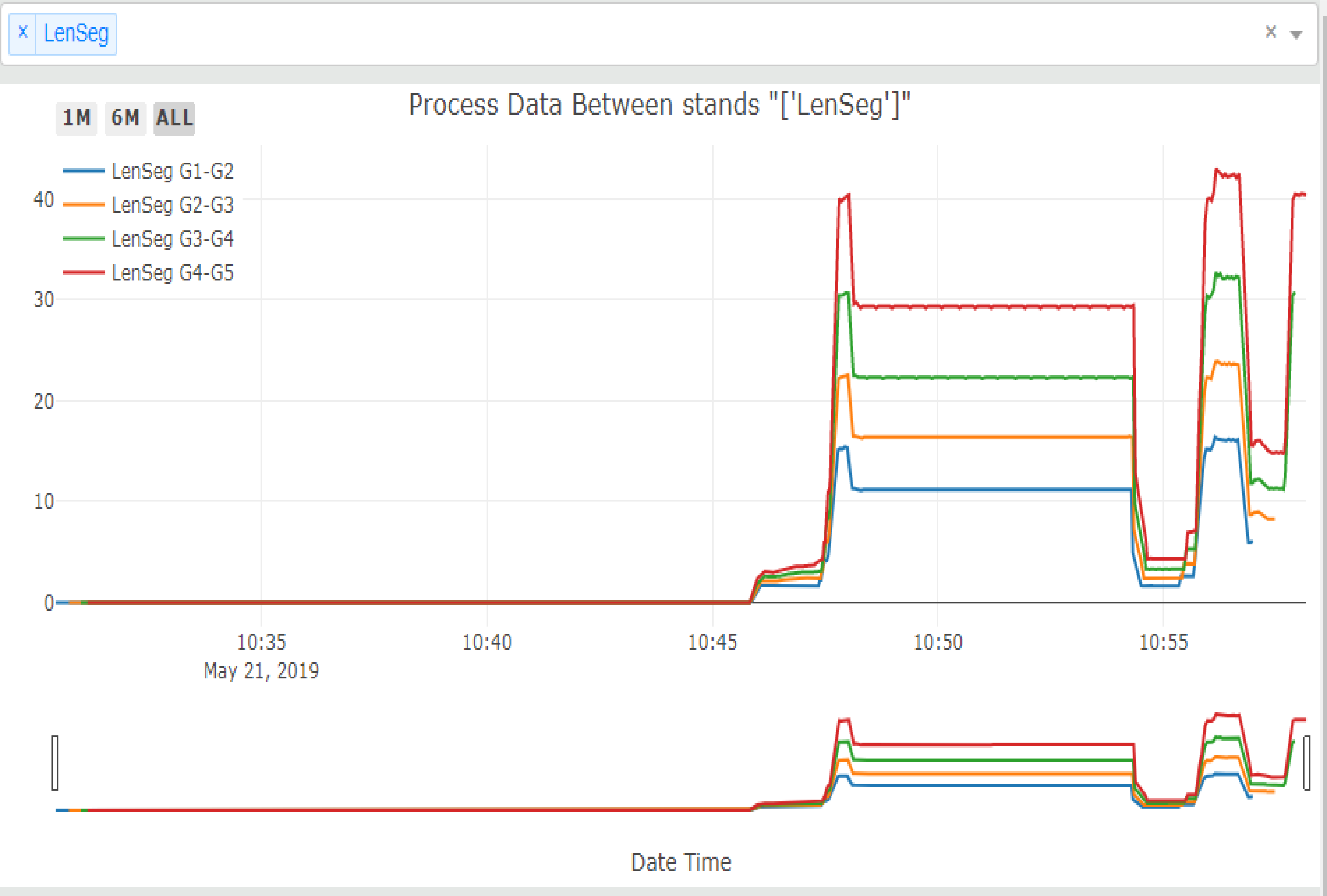
× StripThick

Process Data Before First stands "['StripThick']"

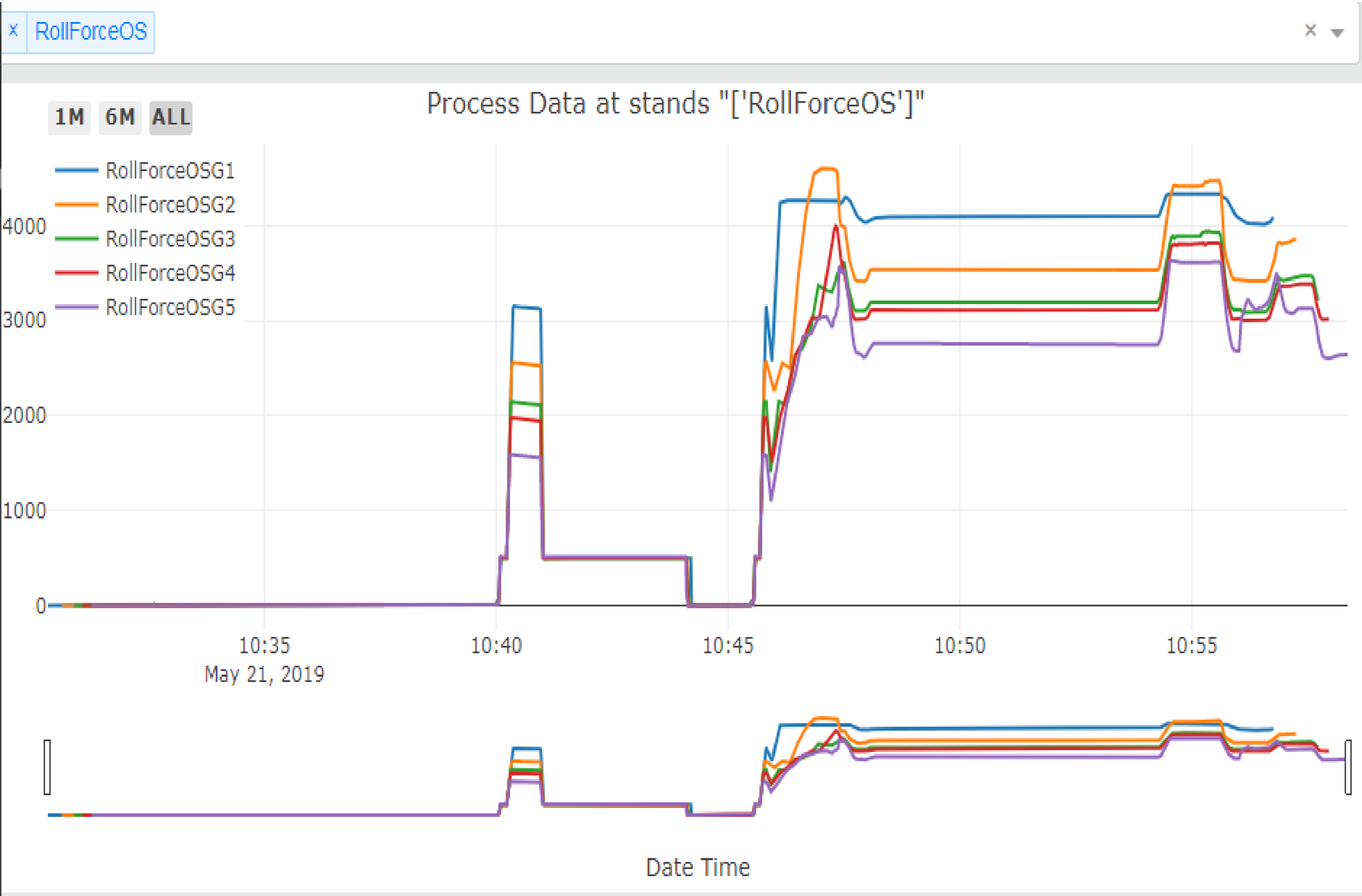
Process Data After Last stands "['StripThick']"



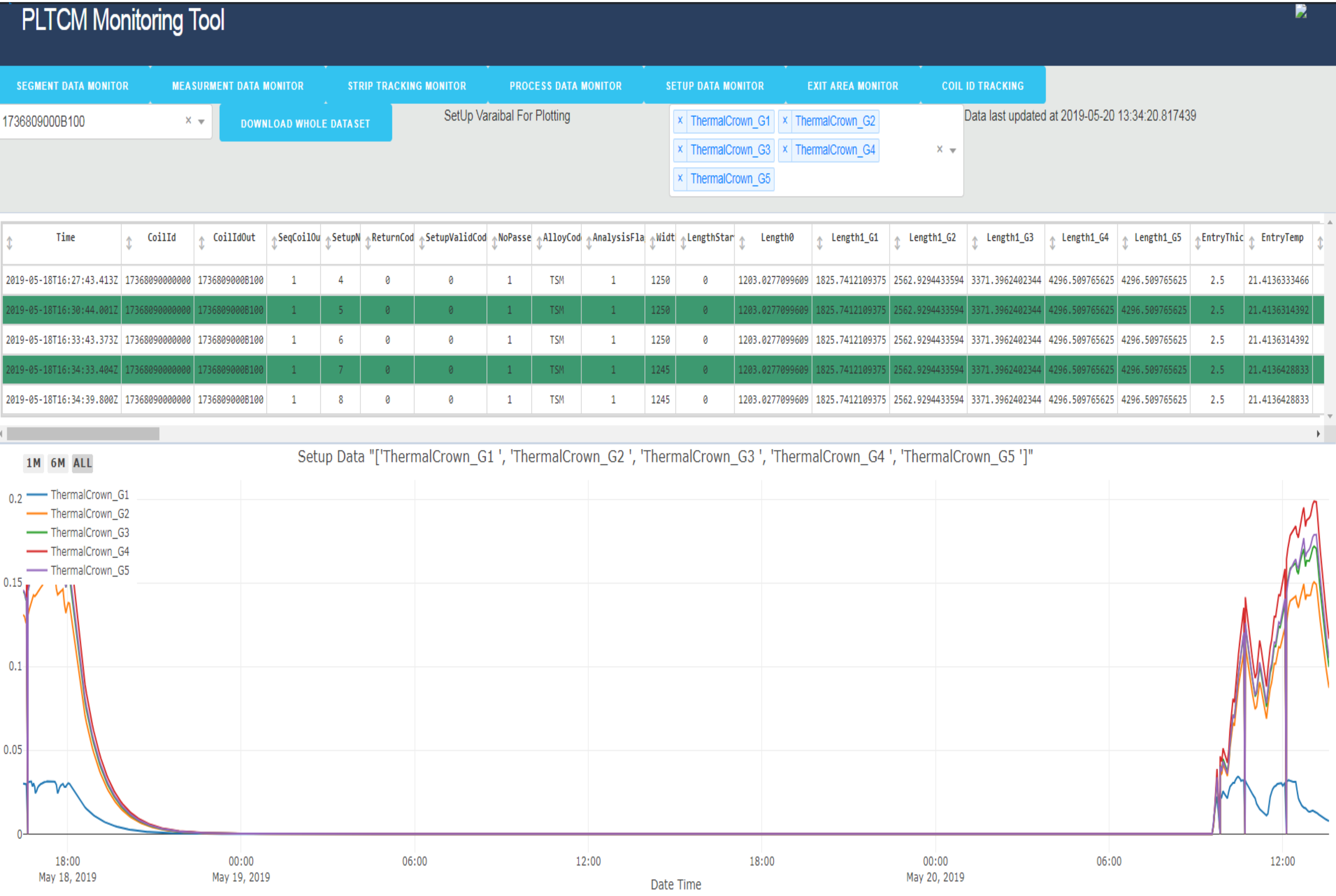
Process Data at stands "['SegId']"



Process Data Between stands "['LenSeg']"



× RollForceOS
×
1M
6M
ALL
Process Data at stands "['RollForceOS']"
RollForceOSG1
RollForceOSG2
RollForceOSG3
RollForceOSG4
RollForceOSG5
4000
3000
2000
1000
0
10:35
May 21, 2019
10:40
10:45
10:50
10:55
Date Time



1736809000B100

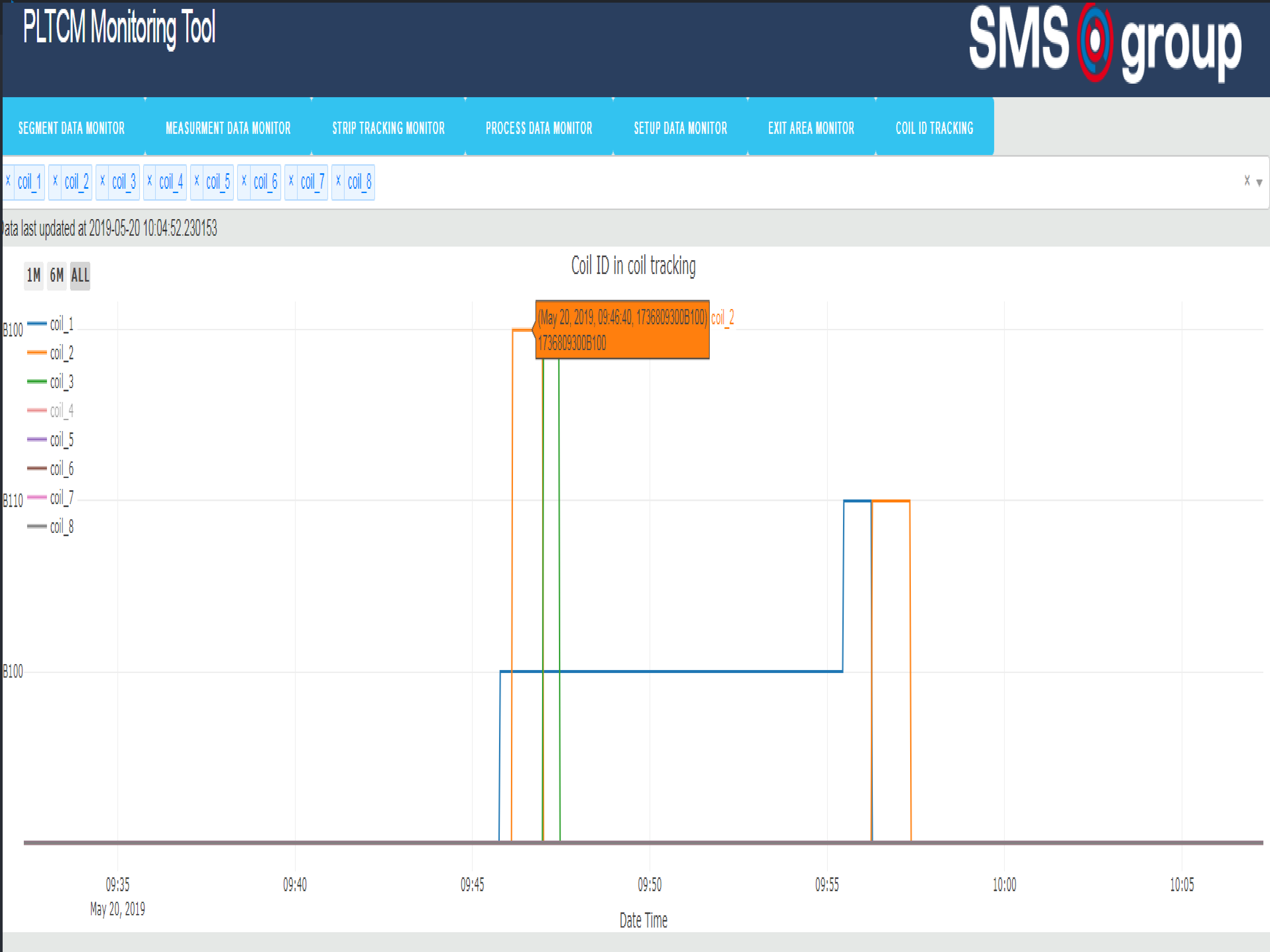
SetUp Varaibal For Plotting

× ThermalCrown_G1 × ThermalCrown_G2 × ThermalCrown_G3 × ThermalCrown_G4 × ThermalCrown_G5

Data last updated at 2019-05-20 13:34:20.817439

Time	CoilId	CoilIdOut	SeqCoilOu	SetupN	ReturnCod	SetupValidCod	NoPasse	AlloyCod	AnalysisFla	Widt	LengthStar	Length0	Length1_G1	Length1_G2	Length1_G3	Length1_G4	Length1_G5	EntryThic	EntryTemp
2019-05-18T16:27:43.413Z	17368090000000	1736809000B100	1	4	0	0	1	TSM	1	1250	0	1203.0277099609	1825.7412109375	2562.9294433594	3371.3962402344	4296.509765625	4296.509765625	2.5	21.4136333466
2019-05-18T16:30:44.001Z	17368090000000	1736809000B100	1	5	0	0	1	TSM	1	1250	0	1203.0277099609	1825.7412109375	2562.9294433594	3371.3962402344	4296.509765625	4296.509765625	2.5	21.4136314392
2019-05-18T16:33:43.373Z	17368090000000	1736809000B100	1	6	0	0	1	TSM	1	1250	0	1203.0277099609	1825.7412109375	2562.9294433594	3371.3962402344	4296.509765625	4296.509765625	2.5	21.4136314392
2019-05-18T16:34:33.404Z	17368090000000	1736809000B100	1	7	0	0	1	TSM	1	1245	0	1203.0277099609	1825.7412109375	2562.9294433594	3371.3962402344	4296.509765625	4296.509765625	2.5	21.4136428833
2019-05-18T16:34:39.800Z	17368090000000	1736809000B100	1	8	0	0	1	TSM	1	1245	0	1203.0277099609	1825.7412109375	2562.9294433594	3371.3962402344	4296.509765625	4296.509765625	2.5	21.4136428833

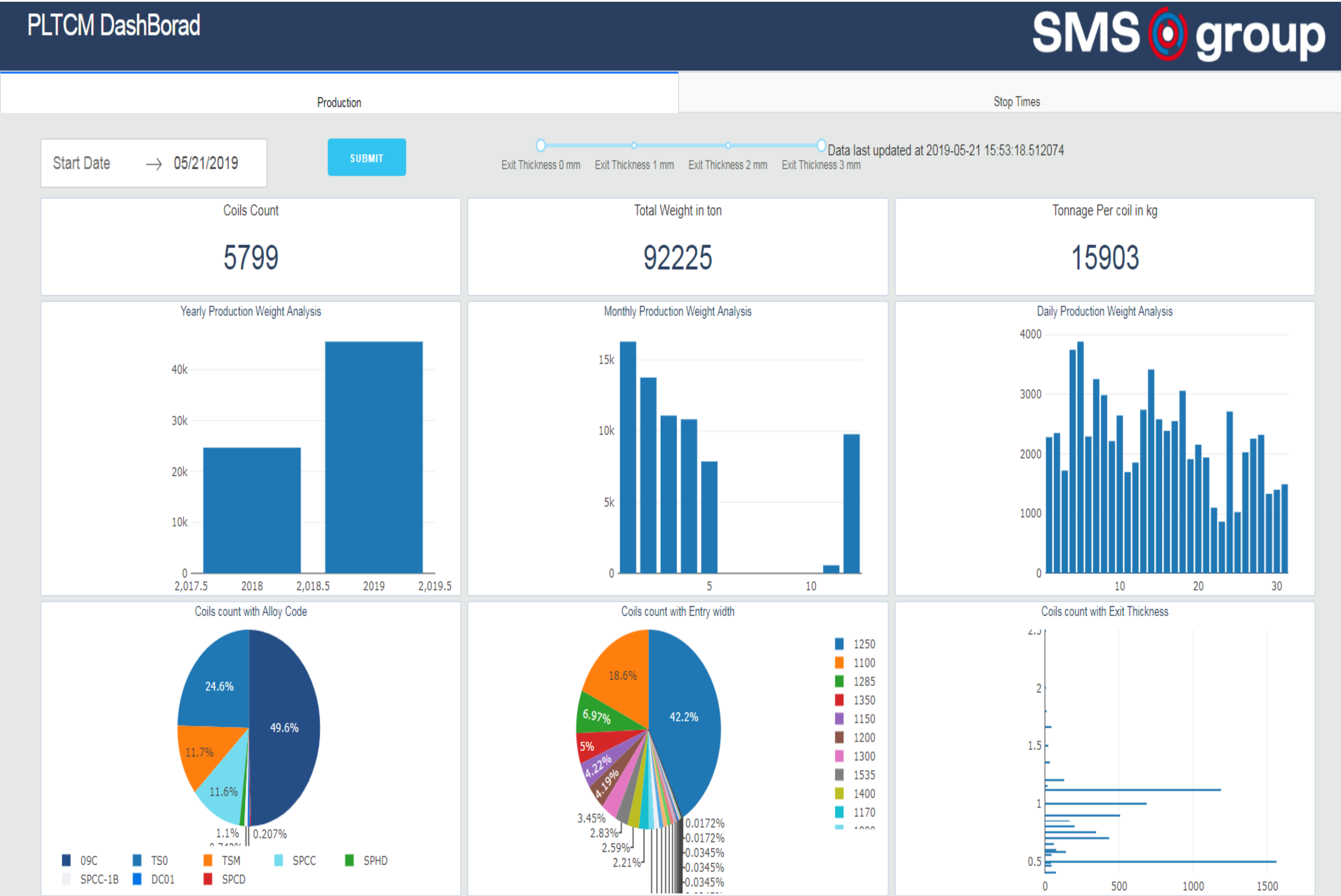
Setup Data "['ThermalCrown_G1 ', 'ThermalCrown_G2 ', 'ThermalCrown_G3 ', 'ThermalCrown_G4 ', 'ThermalCrown_G5 ']"



ata last updated at 2019-05-20 10:04:52.230153

(May 20, 2019, 09:46:40, 1736809300B100) coil_2
1736809300B100

Prototype of Production DashBoard



PLTCM DashBorad

Production | Stop Times

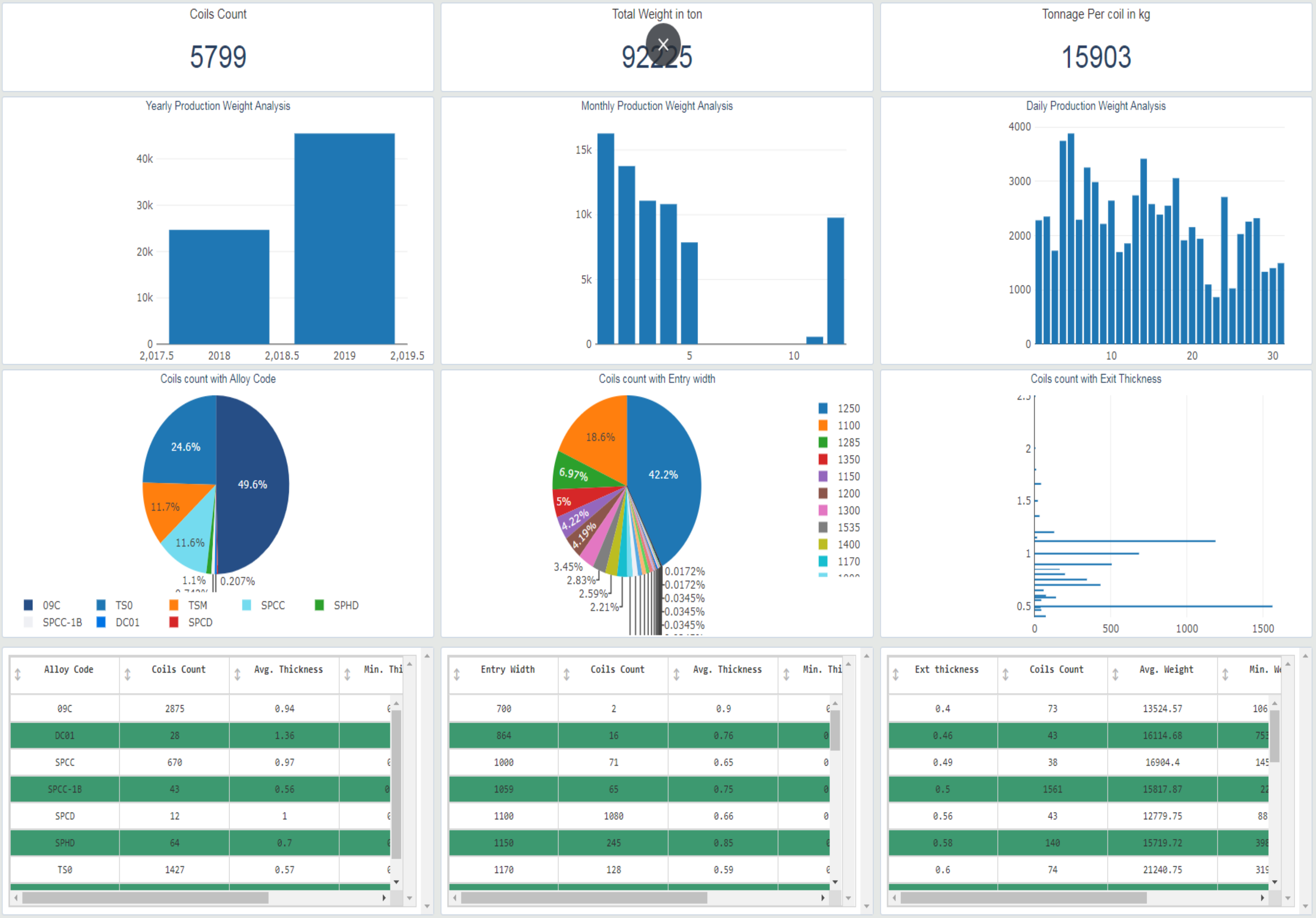
Start Date → 05/21/2019

SUBMIT

Exit Thickness 0 mm · Exit Thickness 1 mm · Exit Thickness 2 mm · Exit Thickness 3 mm

Data last updated at 2019-05-21 15:53:18.512074

Coils Count	Total Weight in ton	Tonnage Per coil in kg
5799	92225	15903

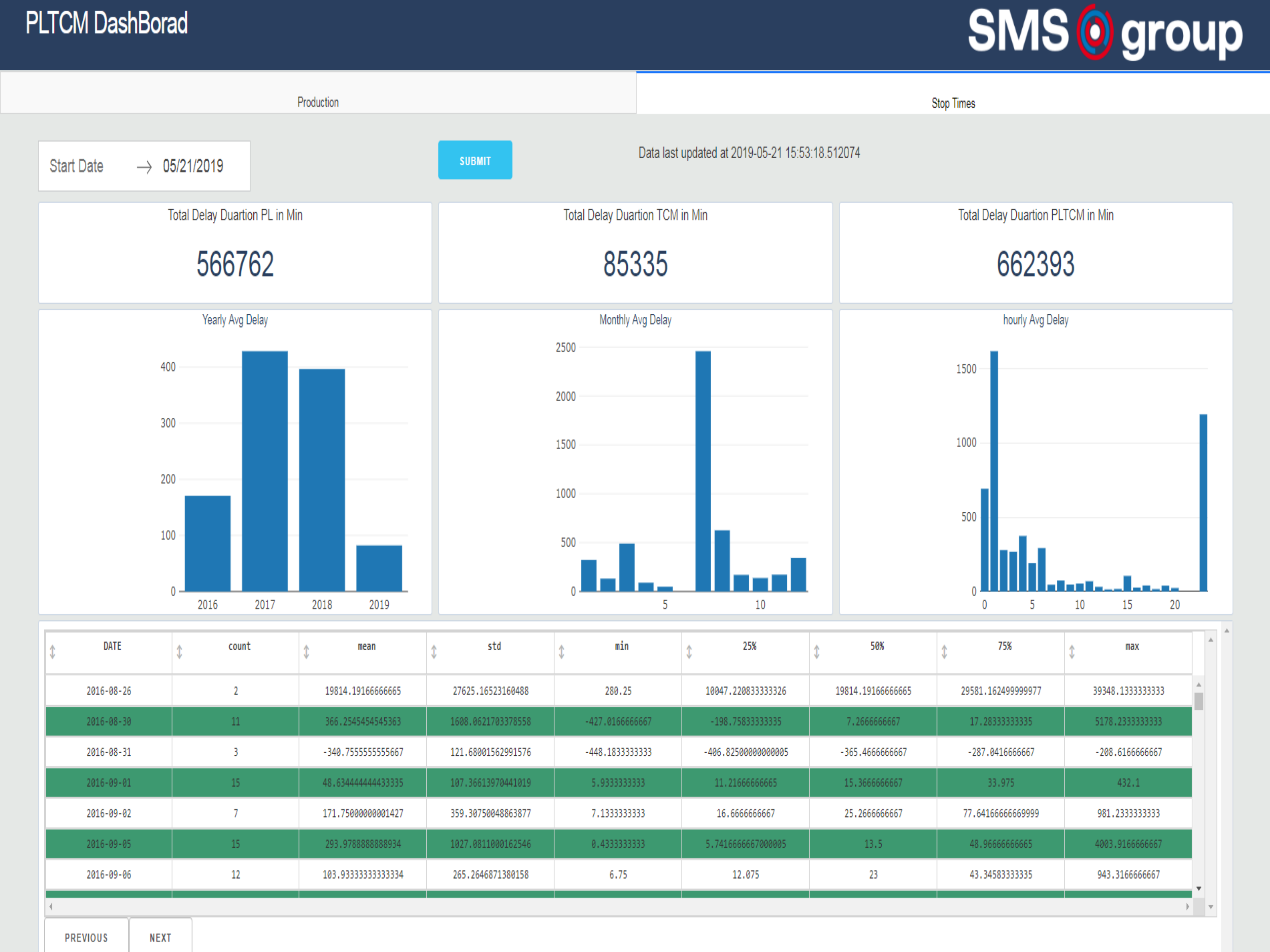


Coils Count	Total Weight in ton	Tonnage Per coil in kg
5799	92225	15903

Alloy Code	Coils Count	Avg. Thickness	Min. Thi
09C	2875	0.94	0
DC01	28	1.36	
SPCC	670	0.97	0
SPCC-1B	43	0.56	0
SPCD	12	1	0
SPHD	64	0.7	0
TS0	1427	0.57	0

Entry Width	Coils Count	Avg. Thickness	Min. Thi
700	2	0.9	0
864	16	0.76	0
1000	71	0.65	0
1059	65	0.75	0
1100	1080	0.66	0
1150	245	0.85	0
1170	128	0.59	0

Ext thickness	Coils Count	Avg. Weight	Min. We
0.4	73	13524.57	106
0.46	43	16114.68	753
0.49	38	16904.4	145
0.5	1561	15817.87	22
0.56	43	12779.75	88
0.58	140	15719.72	398
0.6	74	21240.75	319



Start Date → 05/21/2019

Data last updated at 2019-05-21 15:53:18.512074

Total Delay Duartion PL in Min

566762

Total Delay Duartion TCM in Min

85335

Total Delay Duartion PLTCM in Min

662393

Yearly Avg Delay

Monthly Avg Delay

hourly Avg Delay

DATE	count	mean	std	min	25%	50%	75%	max
2016-08-26	2	19814.19166666665	27625.16523160488	280.25	10047.220833333326	19814.19166666665	29581.162499999977	39348.1333333333
2016-08-30	11	366.2545454545363	1608.0621703378558	-427.0166666667	-198.75833333335	7.2666666667	17.28333333335	5178.2333333333
2016-08-31	3	-340.7555555555667	121.68001562991576	-448.1833333333	-406.82500000000005	-365.4666666667	-287.0416666667	-208.6166666667
2016-09-01	15	48.634444444433335	107.36613970441019	5.9333333333	11.21666666665	15.3666666667	33.975	432.1
2016-09-02	7	171.75000000001427	359.30750048863877	7.1333333333	16.6666666667	25.2666666667	77.64166666669999	981.2333333333
2016-09-05	15	293.9788888888934	1027.0811000162546	0.4333333333	5.7416666667000005	13.5	48.96666666665	4003.9166666667
2016-09-06	12	103.93333333333334	265.2646871380158	6.75	12.075	23	43.34583333335	943.3166666667