
ASTRON
Netherlands Institute for Radio Astronomy

The one class decorator to monitor them all

ASTRON
Netherlands Institute for Radio Astronomy

Corne Lukken (Dantali0n, PD3SU)

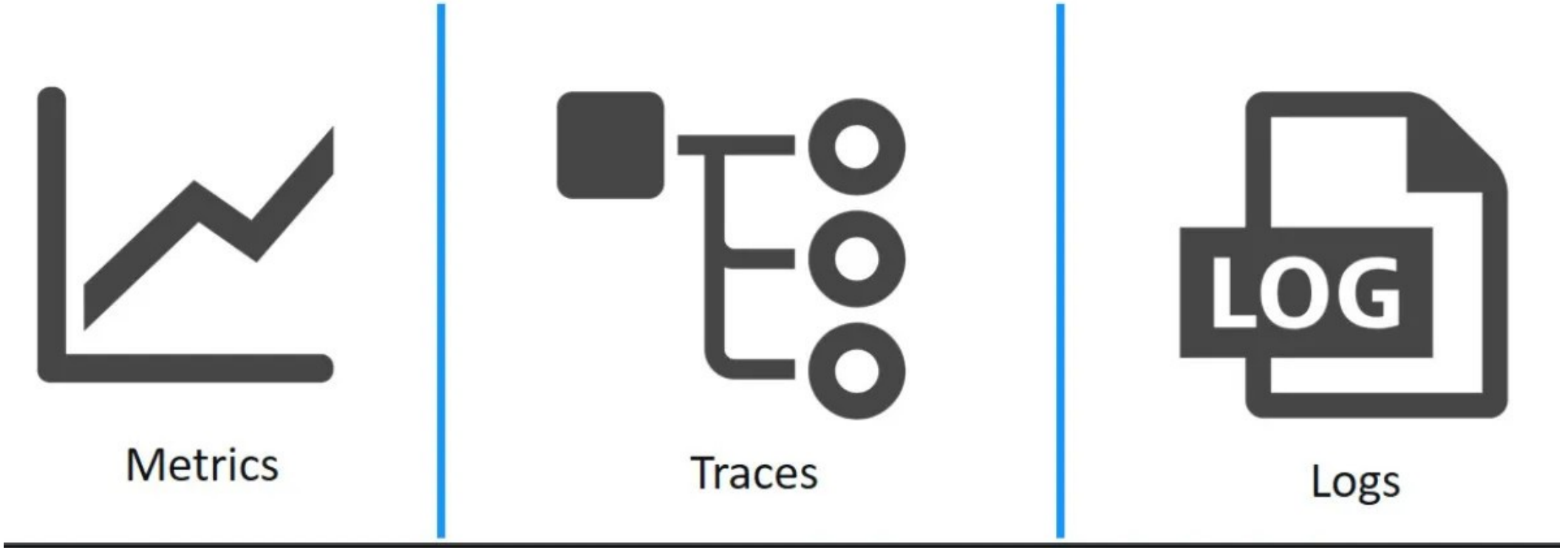
Contents

- Observability
- Prometheus
  - Values, metrics, labels
  - Metric types
  - Handling strings
- The decorator
  - Usage
  - How it works
  - Why?
- Conclusion

https://gitlab.com/ska-telescope/ska-tango-exporter

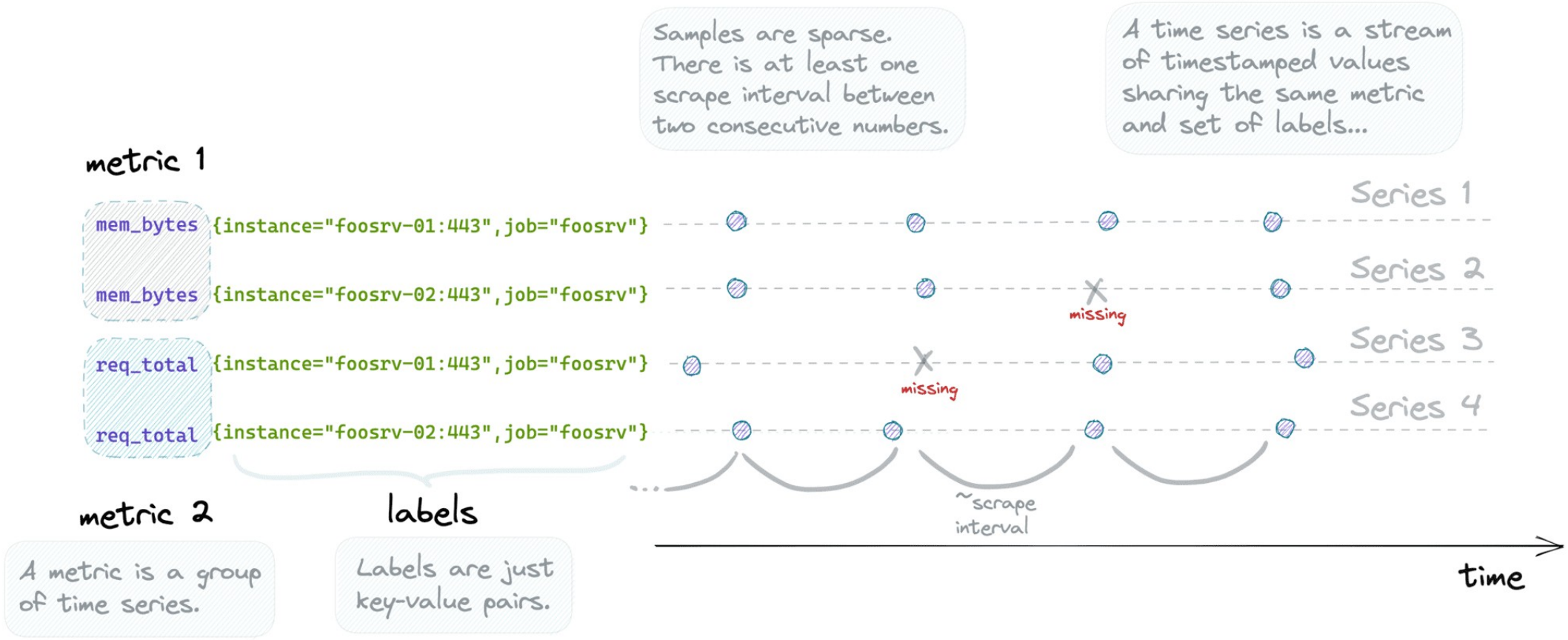
ASTRON
Netherlands Institute for Radio Astronomy

Observability



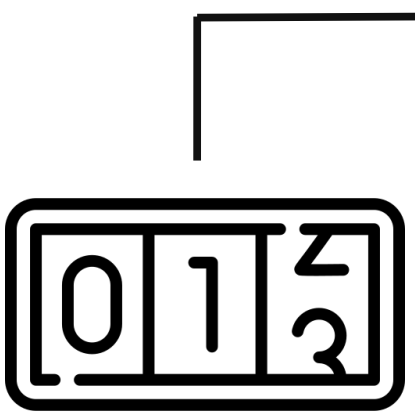
ASTRON
Netherlands Institute for Radio Astronomy

Prometheus

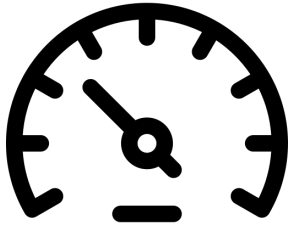


https://iximiuz.com/en/posts/prometheus-metrics-labels-time-series/

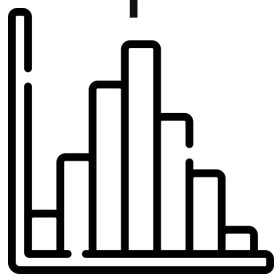
Prometheus - Metric Types



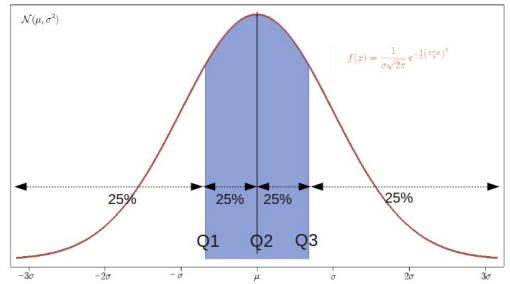
Counter



Gauge



Histogram



Summary

ASTRON
Netherlands Institute for Radio Astronomy

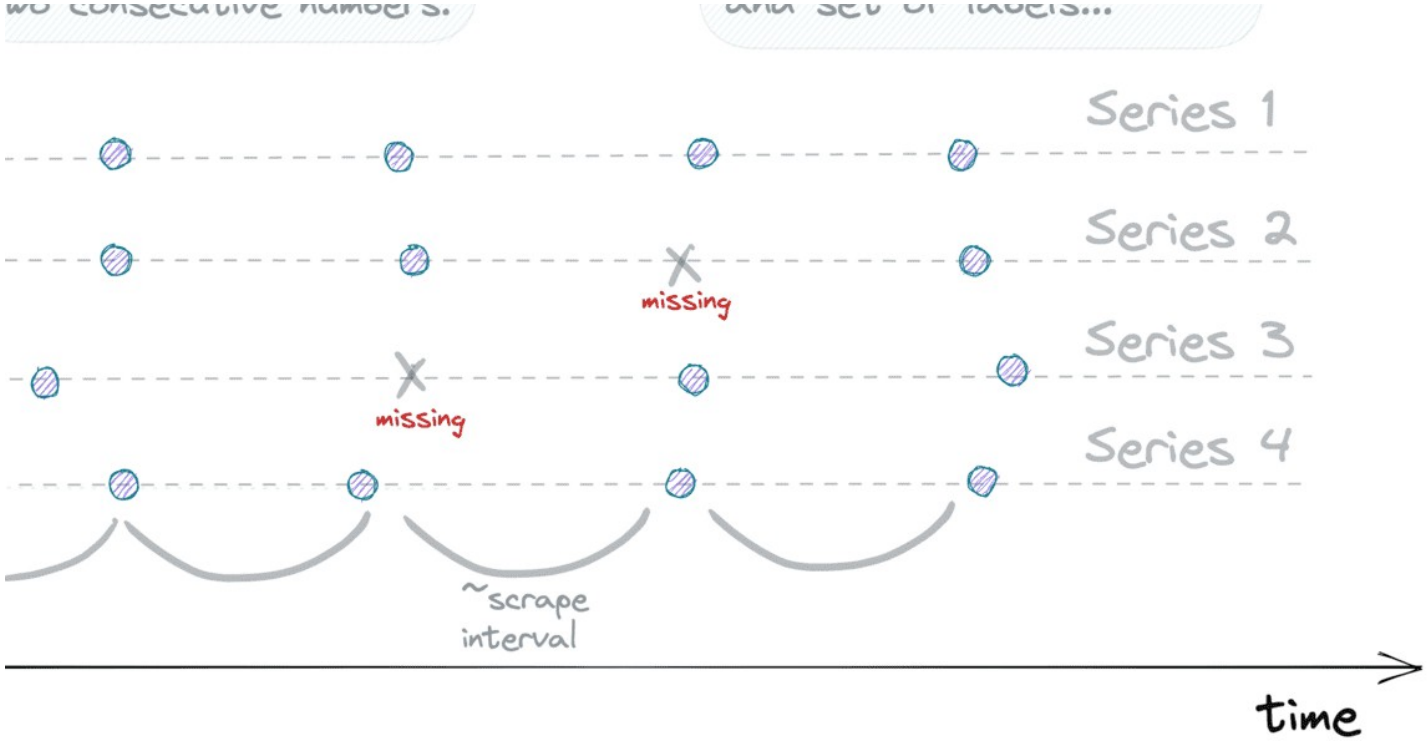
Prometheus – Handling Strings

ds_software_version_: { str_value="23.0.1" }

ds_software_version_: { str_value="18.0.1" }

ds_software_version_: { str_value="19.3.2" }

ds_software_version_: { str_value="21.0.0" }



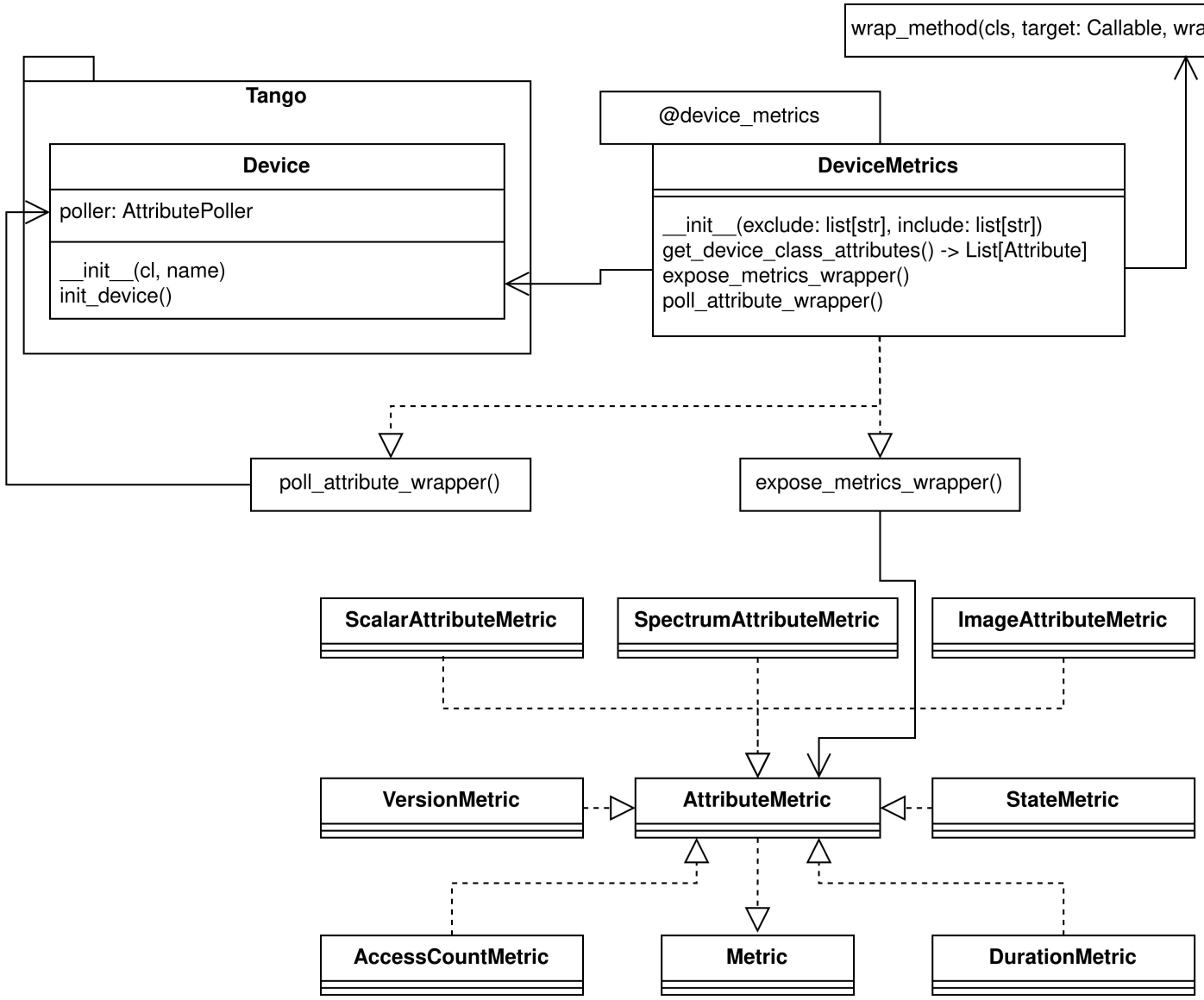
The decorator – How to use

```
@device_metrics(
    exclude=[
        "APSCT_error_R",  # too expensive as they read a lot from hardware
        "APSCT_TEMP_error_R",  # too expensive as they read a lot from hardware
        "APSCT_VOUT_error_R",  # too expensive as they read a lot from hardware
        "*_RW",
    ]
)
class APSCT(OPCUADevice):
```

The decorator – How to use

```python
from prometheus_client import start_http_server, disable_created_metrics

def start_metrics_server(port: int = 8000):
    disable_created_metrics()
    start_http_server(port)

def main(**kwargs):
    args = sys.argv[1:]
    device_class_str = sys.argv[0]
    device_class = getattr(sys.modules["devices"], device_class_str)

    # Setup metrics server (Prometheus endpoint)
    start_metrics_server()

    # Start the device server
    run((device_class,), args=args, **kwargs)
```

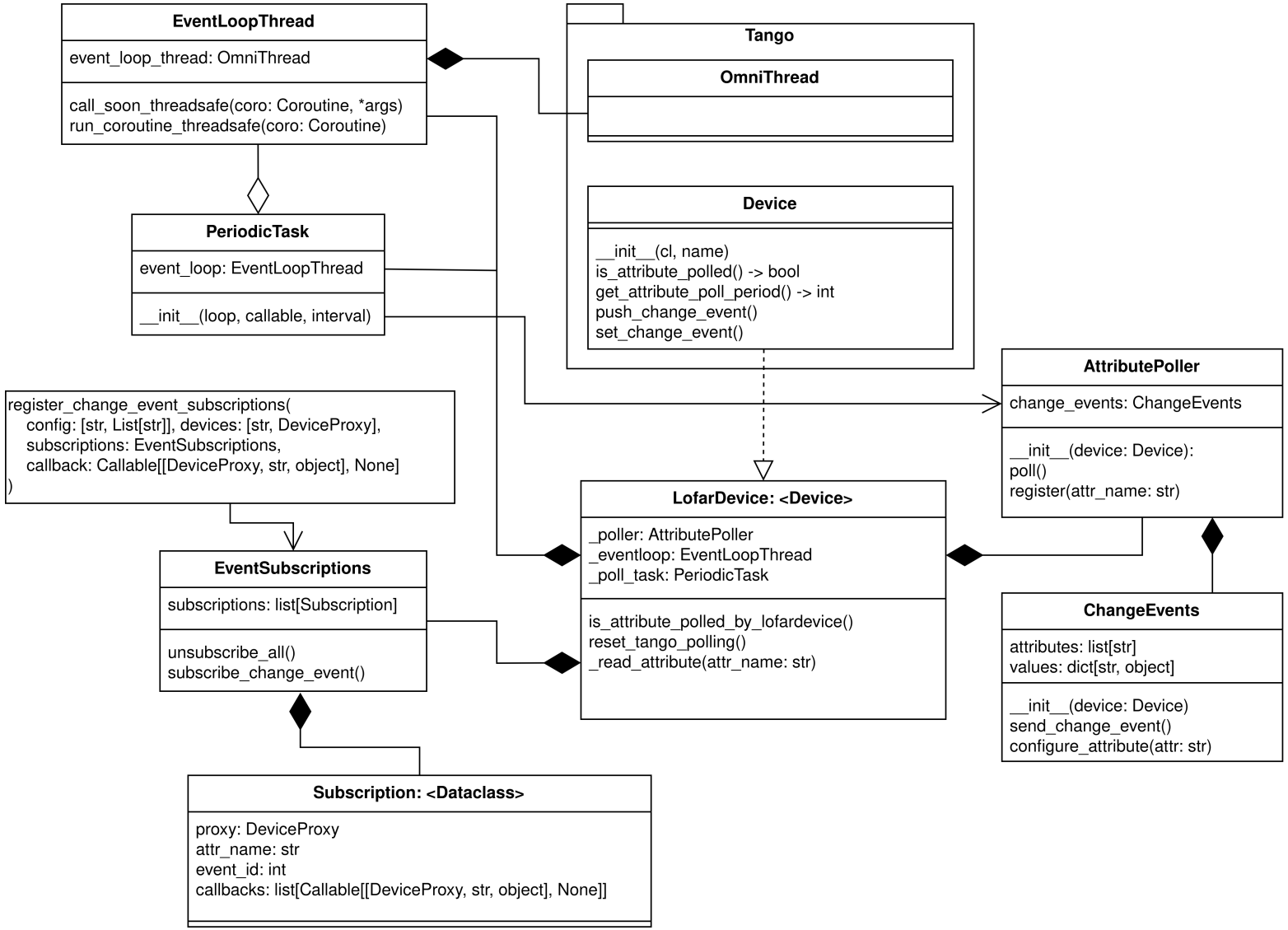
The decorator – How it works




```python
def wrap_method(
    obj: object,
    func: Union[Callable, str],
    wrapper: Callable,
    post_execute: bool = True,
):
    if iscoroutinefunction(func):
        async def pre(instance, *args, **kwargs):
            wrapper(instance, *args, **kwargs)
            return await func(instance, *args, **kwargs)
        async def post(instance, *args, **kwargs):
            return_value = await func(instance, *args, **kwargs)
            wrapper.return_value = return_value
            wrapper(instance, *args, **kwargs)
            return return_value
    else:
        def pre...

    setattr(
        obj,
        func_name,
        functools.update_wrapper(
            post if post_execute else pre,
            func,
        ),
    )
```

The decorator – How it works

The decorator – How it works



ASTRON
Netherlands Institute for Radio Astronomy

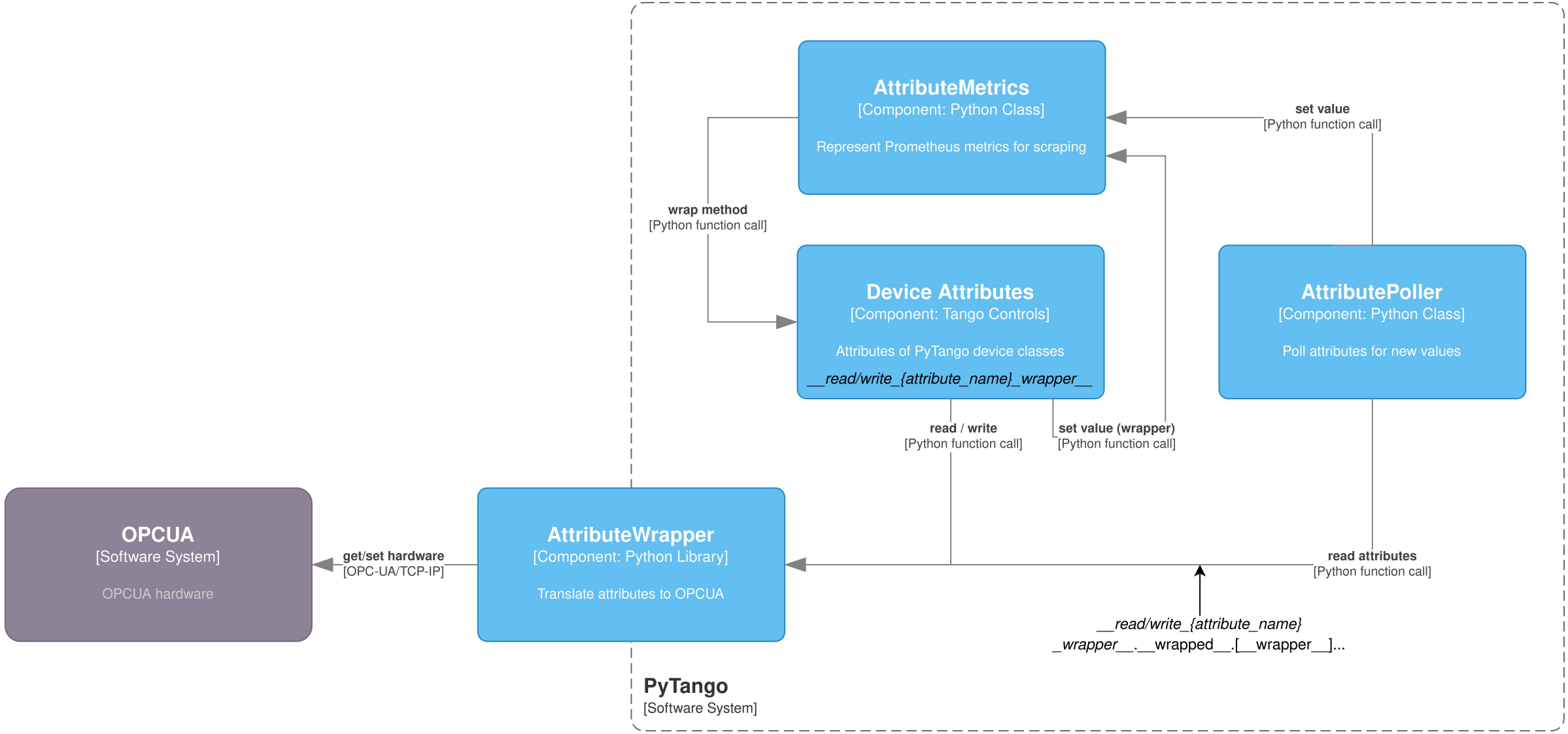
The decorator – How it works

```python
@DurationMetric()
async def _poll(self):
  for attr_name, attr_data in list(self._poll_list.items()):
    if not self.polling_allowed():
      return

    value = await self._read_attribute_nothrow(attr_name)

    if attr_data["metric"]:
      self._update_metric(attr_data["metric"], value)

    ...
        self._send_change_event(attr_name, value)
```

```python
def _read_attribute(self, attr_name: str):
    """Get bound method to read certain attribute."""
    self.get_device_attr().get_attr_by_name(attr_name)

    # obtain the read function (unwrap PyTango's wrapper
    # to avoid interacting with PyTango internals in there).
    read_wrapper = getattr(self, f"__read_{attr_name}_wrapper__")
    while hasattr(read_wrapper, "__wrapped__"):
        read_wrapper = read_wrapper.__wrapped__

    return partial(read_wrapper, self)
```

ASTRON
Netherlands Institute for Radio Astronomy

The decorator – Why?



Conclusion

1) Decorator could use some “native-ication” for Tango
2) Prometheus is a very efficiënt and scalable database for time series numerical data
3) Seen an effective scalable strategy to scraping metrics for PyTango devices
4) Very easy to apply to new devices


Birr
Dutch stations
Chilbolton
Norderstedt
Bałdy
Potsdam
Borówiec
Łazy
Nançay
Medicina
LOFAR
ERIC

The end

source

https://git.astron.nl/lofar2.0/tango/-/tree/master/tangostationcontrol/metrics?ref_type=heads

Thanks to the team!

- Rene Lourens
- Jörn Künsemöller (Universität Bielefeld)
- Jorrit Schaap
- Danita Gnanaraj
- Reinder Kraaij
- Hannes Feldt
- Tom Kamphuis
- Sander ter Veen
- Jan David Mol

ASTRON
Netherlands Institute for Radio Astronomy