Towards a new Sardana Scan Framework:

Generic scan configuration

Jordi Aguilar
Roberto Homs
Oriol Vallcorba
Steven Wohl
at ALBA

Vanessa Silva
at MaxIV

Wojciech Kitka
at S2Innovation

Context

- Requests of new features related with the scan configuration are **pending to be merged**:
  - !2077 *(open, needs work): Optimize motor range in continuous scans to avoid last latency time.* Merged but later reverted😱! The “obvious” fix unintentionally modified latency times.
  - !2085 *(open, needs work): In meshct, prepare measurement group once.* Snake mesh scans not feasible to prepare once😓 without refactoring core MeasurementGroup methods.
  - !2122 *(open, to be reviewed): Irregular MeshCT.* Good to go🚀, but probably introduces logic in CTScan that could be formalized…

Context

- Requests of new features related with the scan configuration are **pending to be merged**.
- Multiple synchronization scans configuration **HLAPI**:
  - Implemented in the core through *MeasurementGroup* API. But no official macros…
  - Missing “adapters” to create multiple synch descriptions from user parameters
    - For example, user specifies synch description per channel, not per synchronizer.
  - Current scans (TScan, CTScan …) are not immediately compatible:
    - Measurement group latency time is not detailed per synchronizer.
    - Theoretical timestamps/positions per synchronizer
    - Recorders are Record-based

Context

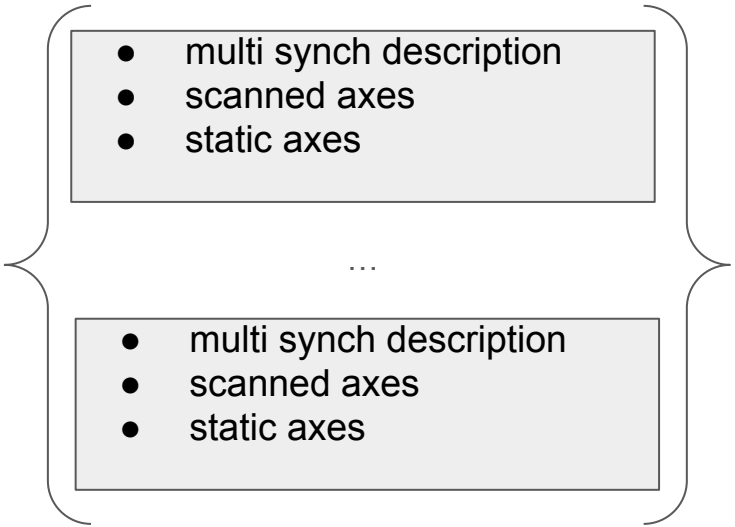
- Requests of new features related with the scan configuration are **pending to be merged**.
- Multiple synchronization scans configuration **HLAPI**.
- **New features** are waiting:
  - Resume scans after interruption (i.e. Ctrl+c)
  - Inter step motor optimization
  - Progress tracking
  - …

Generalize and formalize scan configuration!

- Configuration is scattered in macros…
  - Scans (TScan, CTScan, …) rely on *arbitrary* macro parameters “nb_points”, “integ_time”...
  - In CTScans, waypoints yielded are expected to have specific keys
  - In SScans, steps yielded are expected to have specific keys but not documented

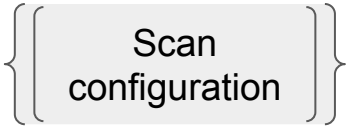
Generalize and formalize scan configuration!

- Configuration is scattered in macros…
- Unify scan configuration (serializable, Pythonic, extendable…)



Generalize and formalize scan configuration

- Configuration is scattered in macros…
- Unify scan configuration
- Build the other Scan Framework blocks around this one



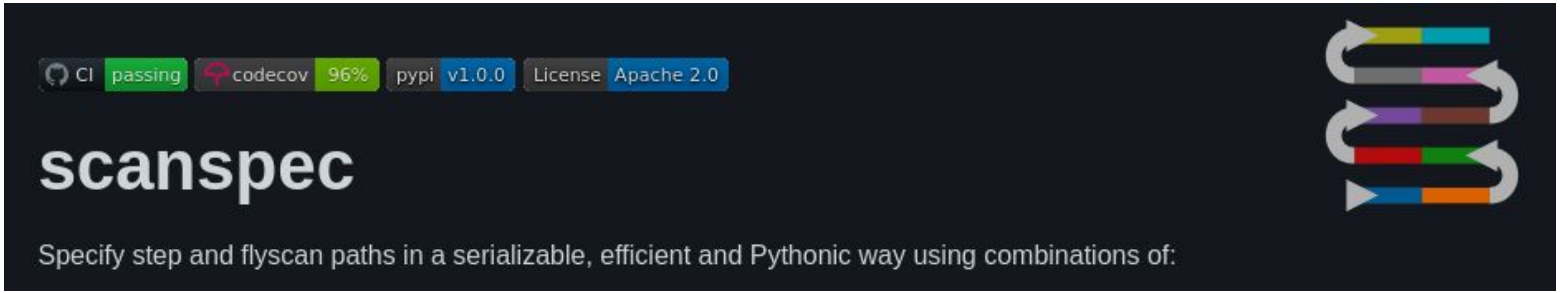
Scan helper:
- Wrap scan configuration building
- Shutter usage
- Add metadata

Scan validator:
- Motor restrictions
- Latency times
- Shutter restrictions

Scan engine:
- Optimize scan
- Prepare mntgrp
- Prepare recorders
- Track progress

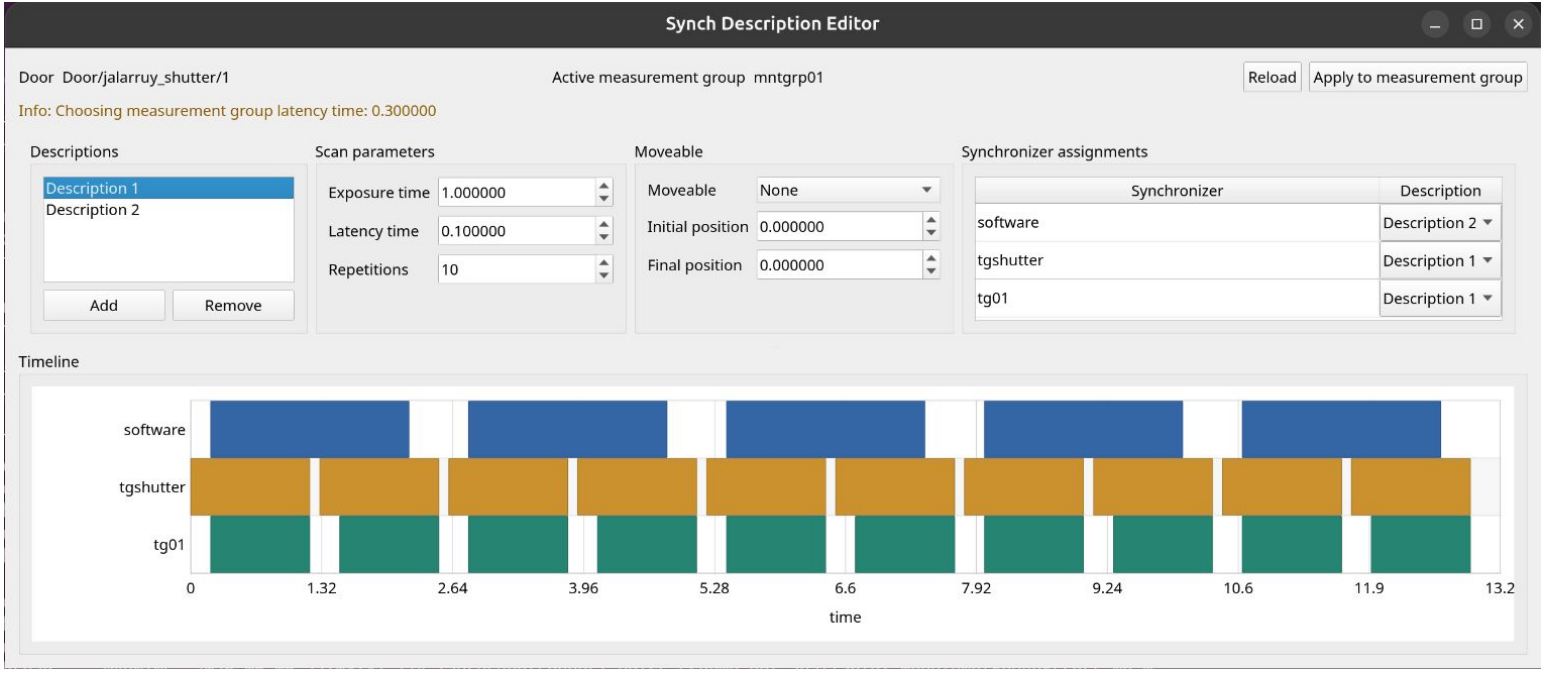
See also...

- Bluesky has its own *scan_spec*, and they are building version 2



See also...

- Bluesky has its own *scan_spec*, and they are building version 2
- Shutter integrated in all Sardana Scans with a *SynchDescriptionHelper*

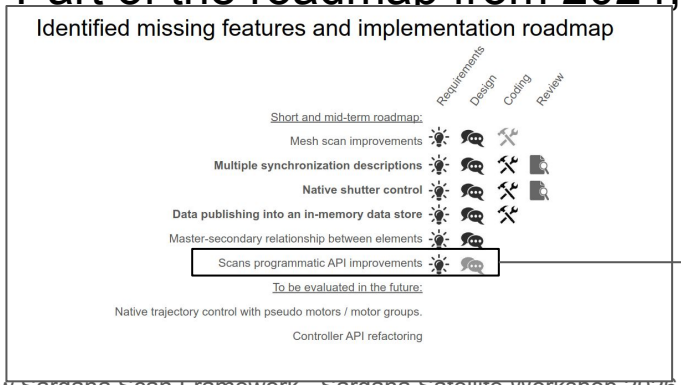


See also...

- Bluesky has its own *scan_spec*, and they are building version 2
- Shutter integrated in all Sardana Scans with a *SynchDescriptionHelper*
- Macros like the ones at ALBA (tomographies) or MAX IV (imeshct, hmoscan) already use json-like configuration. Also CTScan or SScan, only not formalized.

See also...

- Bluesky has its own *scan_spec*, and they are building version 2
- Shutter integrated in all Sardana Scans with a *SynchDescriptionHelper*
- Macros like the ones at ALBA (tomographies) or MAX IV (imeshct, hmoscan) already use json-like configuration. Also CTScan or SScan, only not formalized.
- Part of the roadmap from 2024, more thoughts in !2116



Scans programatic API improvements