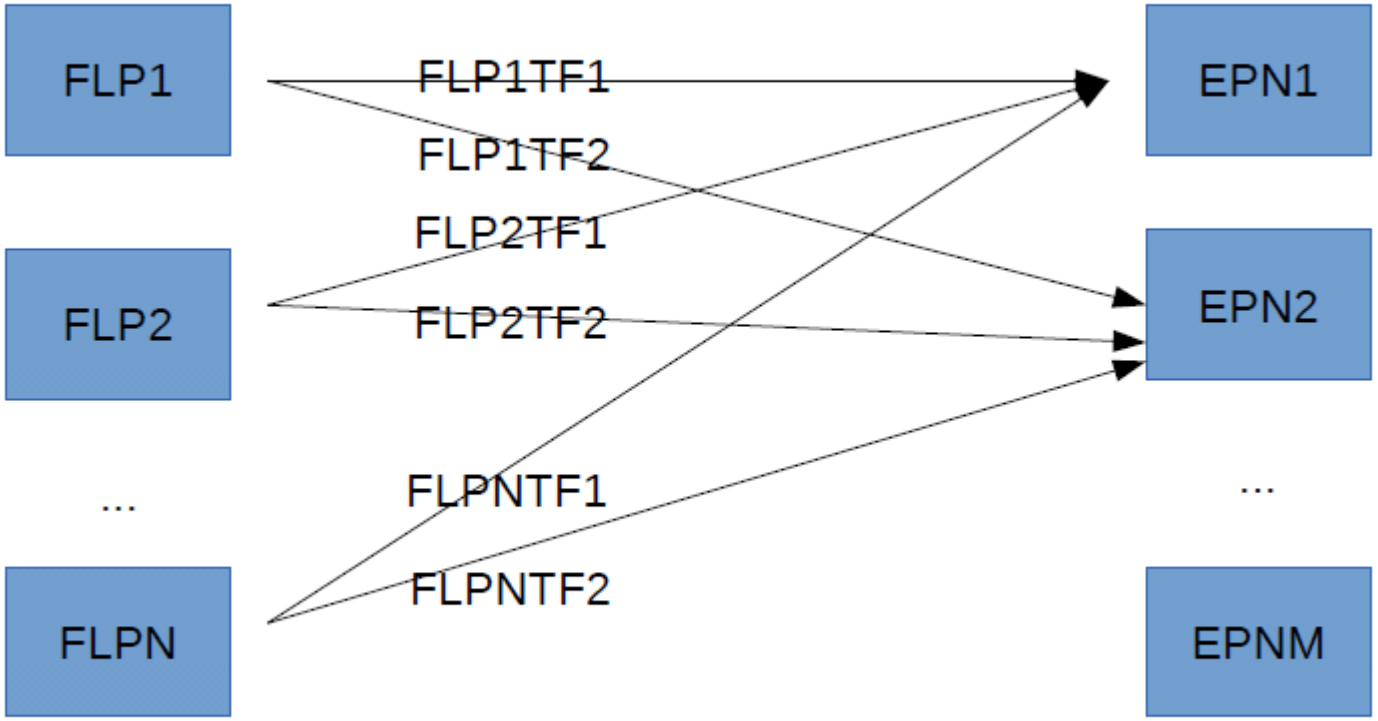
A Large Ion Collider Experiment

FLP to EPN timeframe scheduler

Sylvain Chapeland

Context



Time Frame building: all FLPs have to send the data for a given TF to the same EPN
c.f. Figure 2 page 12 in https://edms.cern.ch/file/1272165/2/ALICE-INT-2013-001.pdf

What we need

- An algorithm on the FLP to know where to ship data for a given timeframe, based on its ID
- This is function F(), as described in:

https://twiki.cern.ch/twiki/pub/ALICE/Cwg3/20131008_cwg3_flp2epn.pdf

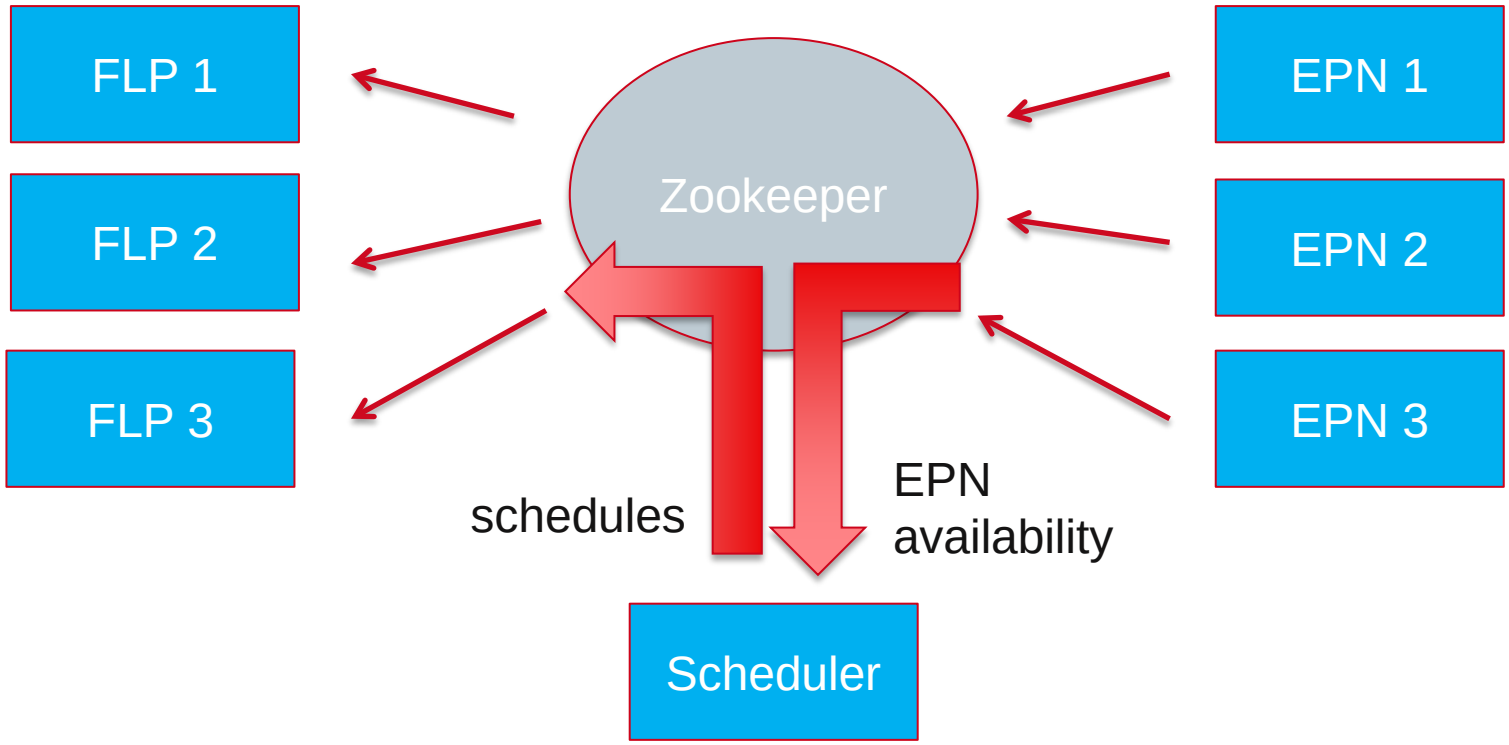
EPN id = F (timeframe id)

- Latency of F() should be minimal
- F() should give the same result on all FLPs
- F() should distribute the load on the available EPNs
- List of EPNs is dynamic (load, failure)

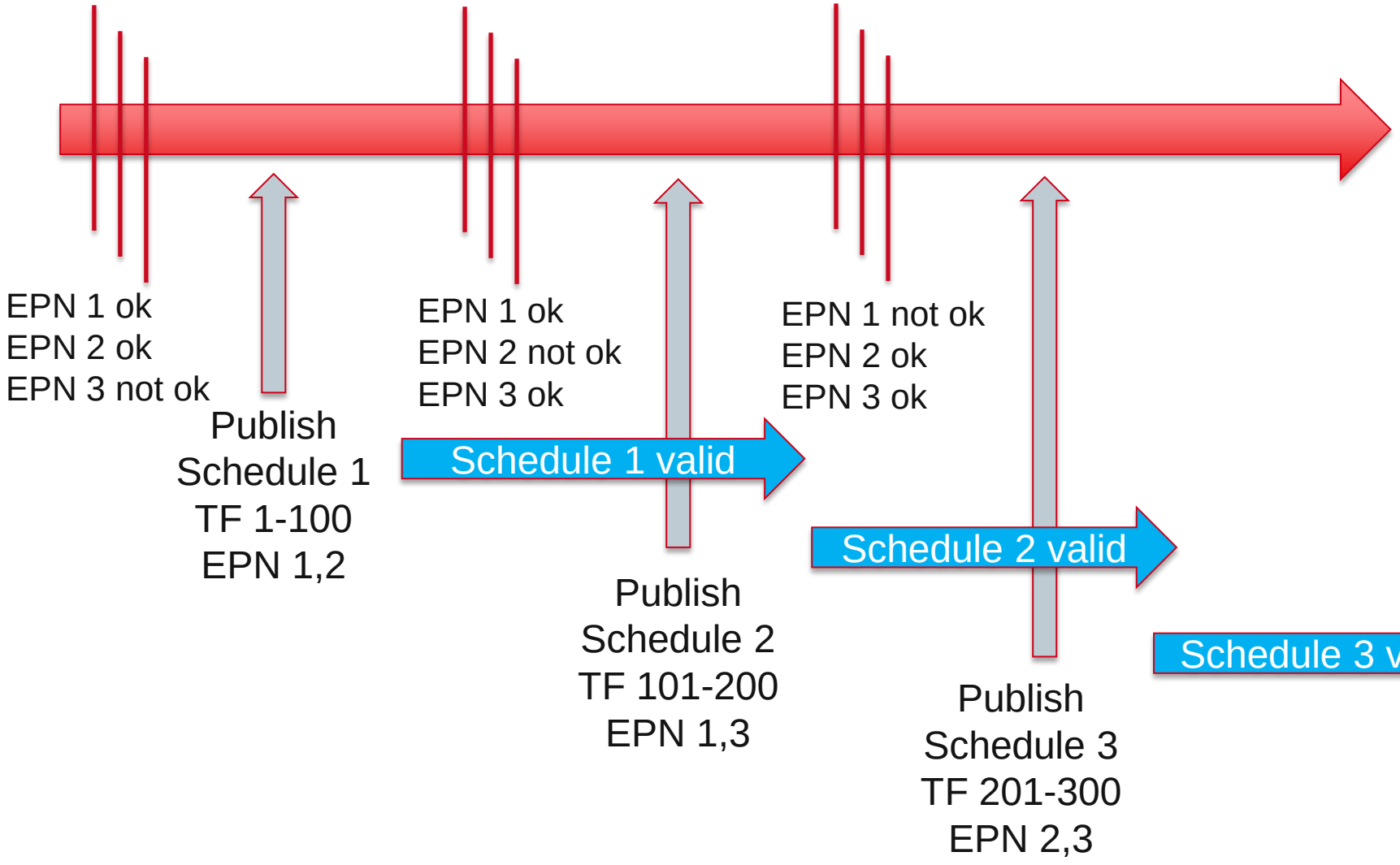
Proposed implementation

- EPNs publish their availability
- A master scheduler process creates and publishes regularly “schedules”:
    - Schedule = list of EPNs to be used for a given range of timeframes
- FLPs subscribe to the schedule
    - F() is a local function using the schedule
- Zookeeper used as (replicated & scalable) information service

Scheduler overview



Timeline


EPN 1 ok
EPN 2 ok
EPN 3 not ok
Publish
Schedule 1
TF 1-100
EPN 1,2
EPN 1 ok
EPN 2 not ok
EPN 3 ok
Schedule 1 valid
Publish
Schedule 2
TF 101-200
EPN 1,3
EPN 1 not ok
EPN 2 ok
EPN 3 ok
Schedule 2 valid
Schedule 3 valid
Publish
Schedule 3
TF 201-300
EPN 2,3

Algorithm

- Schedule published in advance to its validity
  - FLPs publish their max timeframe id (zookeeper)
  - Scheduler dynamically adapts to the rate
  - Schedule range ensured to cover next update interval to avoid FLP blocking

TimeframeToEPN() = listEPN(schedule(TFid)) [TFid % nEPN]

- Typical value:
  - Schedule updated every 5s
  - Schedule published 2s in advance

Interface

FLPs:

```
int getEpnIdFromTimeframeId(
        TimeFrameId timeframeId,
        EpnId &epnId);
```

It is blocking in case schedule not yet published.

EPNs:

```
int setEpnStatus(
        EpnId id,
        EpnStatus status);
```

Code

/afs/cern.mech/user/chapelan/public/epnScheduler.20141106.tar.gz

- Provides C++ API and scheduler process
- Includes dummy FLP/EPN processes for tests

- Interface compliant with O2 coding guidelines (in principle…)
- Embedded Doxygen documentation

Performance

The system is running fine with:

- 250 dummy FLPs (100 Hz timeframes)
- 1250 dummy EPNs (random availability)
- epnMaster configured with refresh 5s and delay 2s (default values).
- Single zookeeper server
- All processes on single machine.
- Latency of F():
  - does not depend on event rate
  - measured <10 microseconds

What's next?

- Integration with CWG13 prototype
  - include code in Git
  - use the code

- Update according to needs and feedback
- Try on a multi-node setup
- Test zookeeper replication