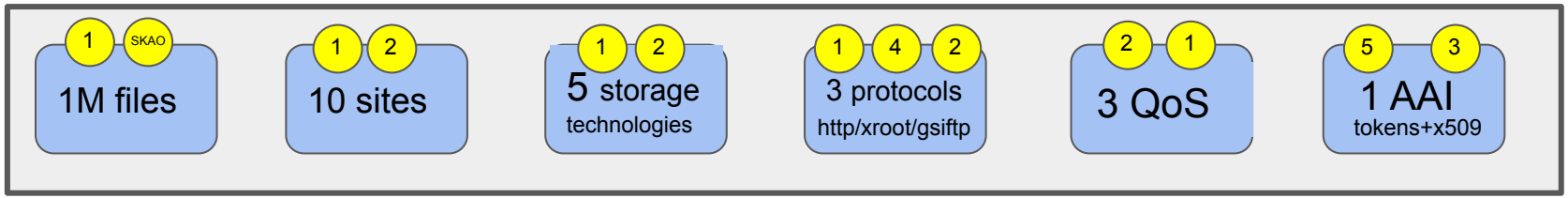
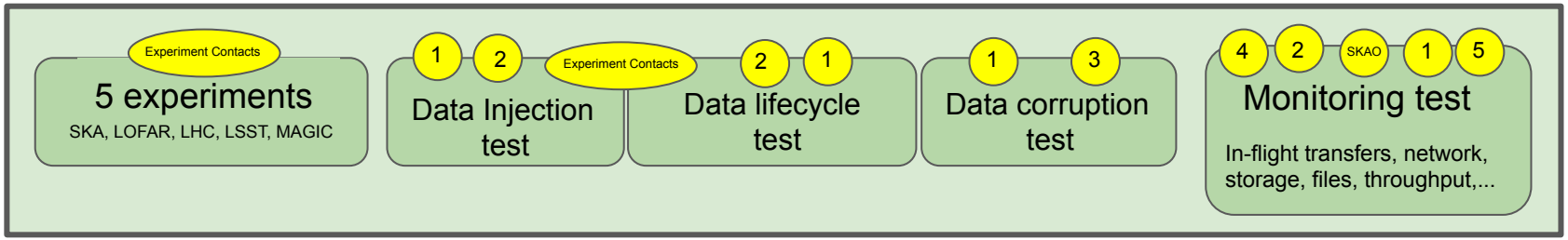
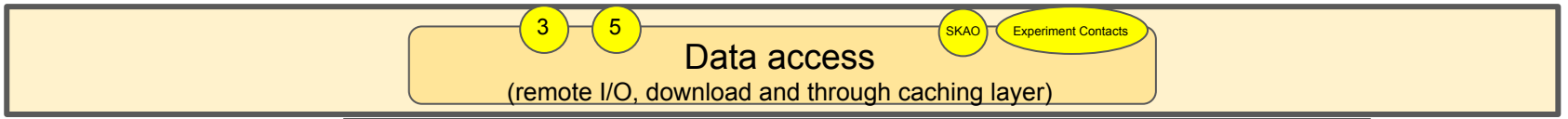
“24h” pilot datalake performance assessment

Goal: Exercise covering **experiment data workflow** needs on a single day. From data injection, to data replication and data access. Three fold goal: perspective from **scientists**, perspective from **sites**, and the assessment of the **ESCAPE datalake tools and services** under **pseudo-prod conditions**: RUCIO, FTS, CRIC, IAM, PerfSONAR, monitoring, QoS, clients, etc.

Work plan: Sept-Oct preparations and tests of the different components in order to run together on a single day, ‘a la’ **dress rehearsal**, mid-November (first challenge, probably a 2nd go afterwards before Xmas break)







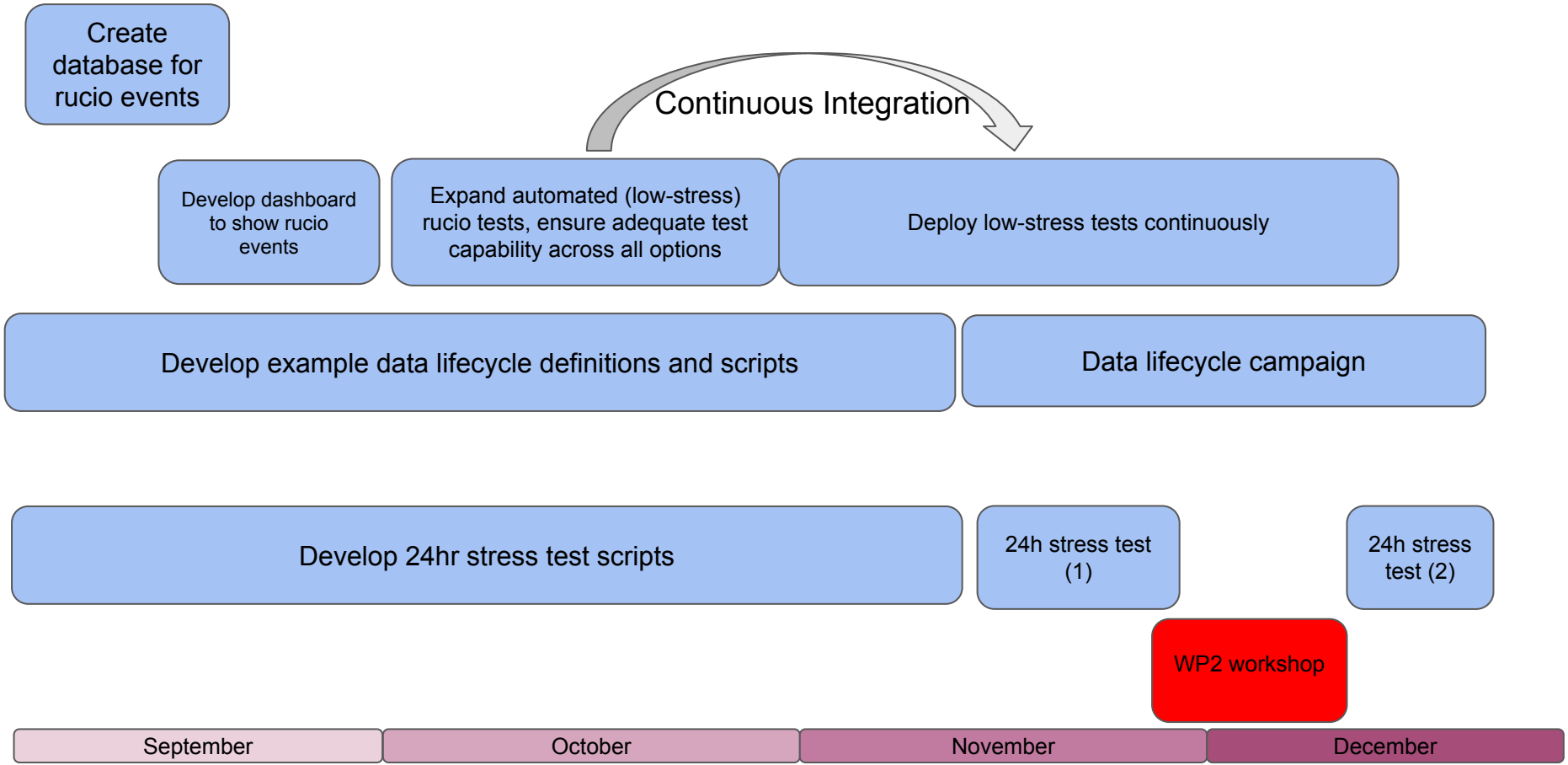
1- datalake 2- QoS 3- integration with compute 4- Networking 5 -AAI

Developing a testing suite

We have already good access to dashboards displaying results from network health monitoring (perfSONAR) and the FTS transfers

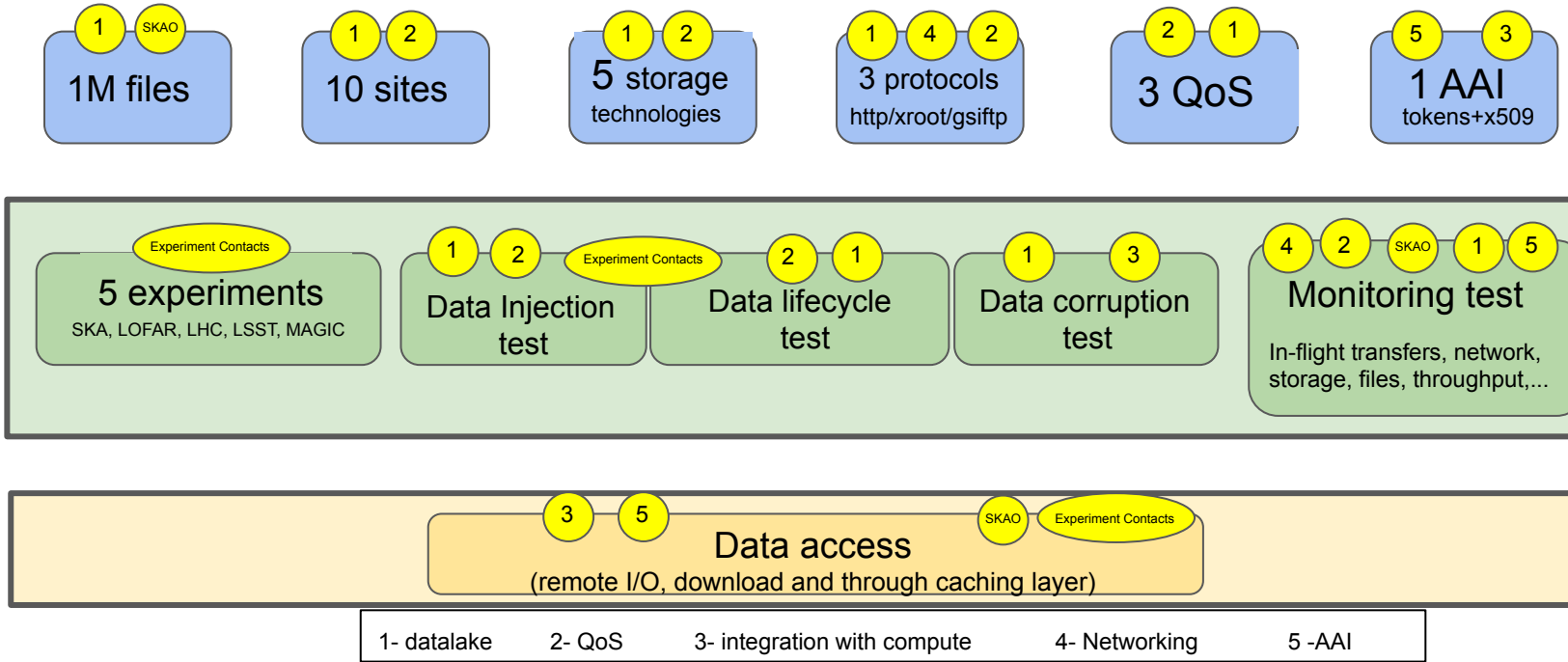
- To assess the datalake we need to get rucio events into a dashboard too
  - Urgent work, to be completed by end September, starting immediately
- Develop and run rucio-level tests
  - Use rucio rules, run automatically
  - Test a range of different parameters - file size, file rate, transfer protocol, error recovery, QoS
  - Framework will be the same as for the more intensive tests
  - Build confidence in these tests during Sep-October, before first intensive campaign

Testing roadmap overview



Data Lifecycle assessment: Medium-term campaign pilot datalake performance assessment

Goal: Test experiment-driven workflows that require a longer campaign - e.g. 4-6 weeks, able to test data lifecycles
Work plan:



1. DataLake Objectives and Needs

- 1M files - to demonstrate stable and sizeable data movement
  - Automatise data injection from Rucio Client and/or user GET/PUT/POST
  - DataLake accessible by different clients (e.g. simple HammerCloud job)
  - Test data injection and lifecycle using mock data profiting from already existing tools
  - Weekly tests of injection and deletion towards November performance assessment
  - Data corruption starting with “done-manually”

- 10 Sites, 5 Storages Technologies, 3 Protocols
  - Stable infrastructure
  - Transfer matrix green
  - Automated (FTS) tests
  - Automated gfal tests
  - Noise (Rucio) production from defined RSEs

1. DataLake Objectives and Needs

- Monitoring
  - Test in-flight transfers, storages, etc.
  - Rucio events traced e.g. by scope
  - Site/RSE and transfers/configurations carefully monitored

- 3 QoS
  - Starting simply with A, B, and C QoS
  - CRIC as reference point
  - Changing QoS within a site and across sites (on demand, by policy)

- Preparations for WP2-M2.4 (Mar 2021)
  - HammerCloud ready to run realistic research infrastructure workloads
  - Real data distribution and analysis for non-HEP RI (LOFAR, CTA, LSST, MAGIC)
  - Ability to plug heterogeneous clouds (commercial)

2. QoS Objectives

- Demonstrate compute-driven QoS staging:
  - Data is made available for a limited period at resources “close to” some computing resources with the appropriate QoS for that computational work-flow.
- Demonstrate cost-performance QoS trade-off:
  - Data QoS changes so that it is stored on cheaper media.
  - Triggers: elapsed time, inactivity, embargo period ends, ...
- Demonstrate VO-specific work-flows:
  - Exercise selected work-flows from QoS document
- Demonstrate data injection with targeted QoS:
  - Data is written into the data-lake with a desired QoS.
- Demonstrate computational match-making:
  - Select the best computational resource, based on data locality (excluding data locality within any caching and latency-hiding layer) and desired QoS.

2. QoS: What's missing

- Selecting three (or more) QoS classes.
  - Classes will use the simplest approach: predefined Storage QoS Class names.
- Selecting the VOs and work-flows to exercise.
- Build up framework for exercising QoS/Rucio interactions
  - Automate the Rucio interactions, to build appropriate load
- Building up knowledge on how to do match-making: for data-injection (selecting endpoint) and computational match-making

3. Compute integration objectives

- Interactive access to files in data lake (e.g. using the Jupyter notebook)
  - Interactive processing by users of experimental data.
  - This also includes transparency of AAI (see slides on that)
- Batch access to files in data lake
  - Simple pipeline(s) to process large amount of data, with fixed parameters
  - Reingesting output data
- Access to data on remote data lake locations through caching layer vs getting data locally for processing
- Data corruption, reporting of corrupted data or fixing it is very relevant, but may be done later on in the project

3. Compute integration needs

- Need use cases that are very simple in terms of computing (mostly mimicking the data flow, not the processing itself) which cover the range of small requests, large requests, small files, large files.
  - LOFAR and ATLAS workflows cover some ground already
- Data sets going with the use case
  - Maybe also a form of data hierarchy per project may be a useful concept to have for example for LOFAR observations, calibrators and targets so that data sets can be found together

5. AAI testbed objectives

- **User enrollment flow in place**
  - Users can enroll in the ESCAPE VO using their institutional credentials, apply for group membership and link X.509 certificates to their account
  - Group administrators for each experiment can approve group membership requests
- **X.509/VOMS AuthN/Z in place**
  - Users can get a VOMS credential reflecting their ESCAPE VO membership and can access data in the data lake using their VOMS credential
  - Users can only access data belonging to their experiment from CLI and Web-based applications
    - Group-based VOMS authorization is in place and correctly configured at datalake sites
- **Token-based AuthN/AuthZ in place**
  - Users can authenticate using their institutional credential and get an access token reflecting their ESCAPE VO membership and can access data in the data lake using their access token
  - Users can only access data belonging to their experiment from CLI and Web-based applications
    - Via scope or group-based authorization

5. AAI: What’s missing

- An agreed-upon definition of the ESCAPE data namespace structure
  - To ensure consistent authorization across the data-lake
- Documentation on token handling and usage
  - Also covering integration with higher level services (e.g., from WP5)
- Continuous monitoring tools to assess that AuthN/Z work as expected at sites
  - Probe that gets VOMS proxy/JWT access token from IAM and checks data access
  - This may potentially be sorted out using RUCIO monitoring
- ??