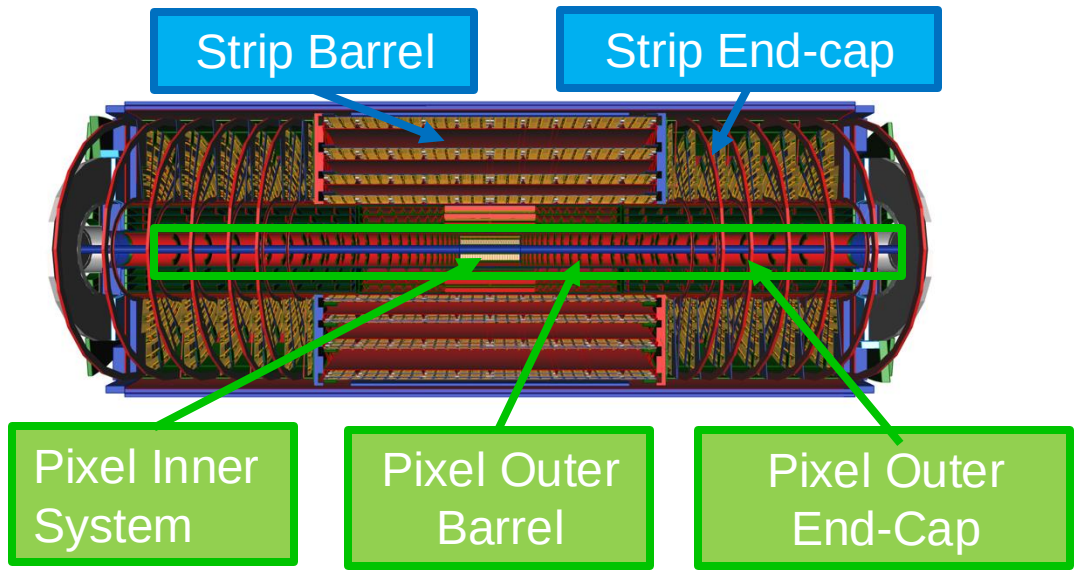
Experience with ITk production database in ATLAS

UK EIC meeting, March 12

Monika Wielers (STFC – RAL)

Overview of the ATLAS ITk Detector

- ITk is the new ATLAS all-silicon inner tracking detector being build for running at the HL-LHC
  - 165 m^2 of Si detector coverage
  - Strips: 4 layers in barrel, 6 disks in fwd/bkd direction
  - Pixels: 5 layers in barrel, up to 18 rings in forward/backward direction
- All parts used in detector (readout chips, sensors, modules, supports, services) to be tracked and QC/QA test results recorded (including prototypes and pre-production items)
- Total production period is ~5 years.
  - ~100 institutes from 22 countries (incl industry partners)
- In total ~10^6 parts to be produced
  - Need for a **database** to store part information, monitor production and allow data-mining in case issues occur during build process or operation



ATL-PHYS-PUB-2021-024

ITk (ID)	Area (m^2)	# Modules	#channels (M)
Pixels	13 (1.6)	~9000 (~2000)	5100 (92)
Strips	165 (61)	~18000 (~4000)	60 (6.3)

New (current) inner detector

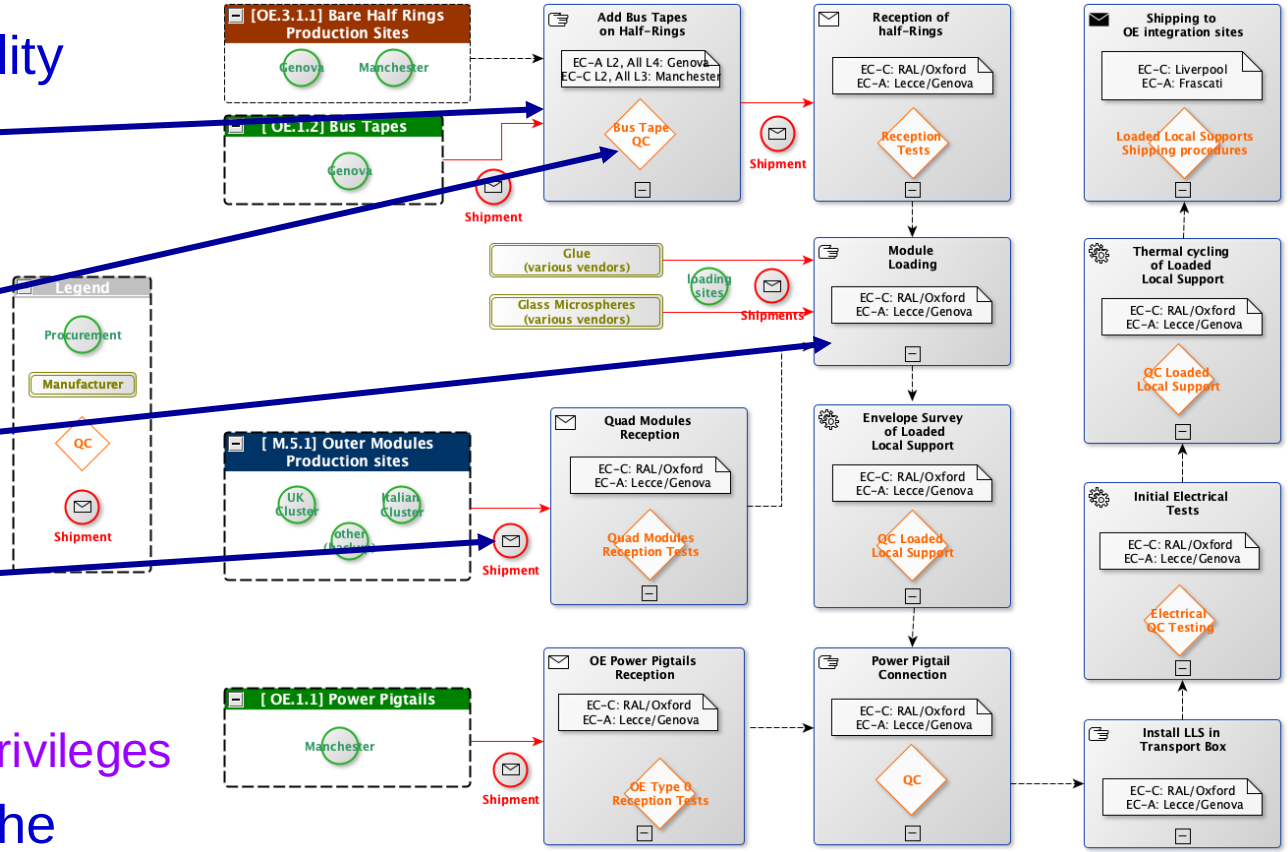
Project	Staves	Modules	Sensors	FE Chips
pixels	354	9464	9464	34292
strips	776	17888	17888	233856

Database Development

- ATLAS opted to have the database developed by commercial company
  - Not enough manpower was available to develop one on our own
    - DB not a delivery for the experiment
- Pros
  - No need to find the manpower to develop one
  - Developed according to standards
- Cons
  - Costs (also includes costs to run the database during detector operation)
  - Cannot use infrastructure in computing centre at laboratory hosting experiment
  - Need to oversee development to ensure product well tailored to your community
    - Our working environment very different from standard companies
- General
  - Need well written requirement document before project starts
    - Includes how uploads are expected to be done and handling the pass/fail test decision
    - Decide what you need being developed by vendor and what by your community, e.g. ask for API commands and expect scripts/web apps for data upload / reporting is done in your community

Production Database: Requirements

- Main production database (PDB) functionality
  - Registering parts (components)
  - Dealing with batches
  - Serial number scheme
  - Tracking build status (stage)
  - Adding test results
  - Handling assemblies
    - Tracking of parent/child relationships
  - Tracking shipments
  - Querying data
  - Assignments of users, institutes and their privileges
- Designed to allow up to 50 users can use the database simultaneously
- Component information to be retained during construction and 10 years of data-taking

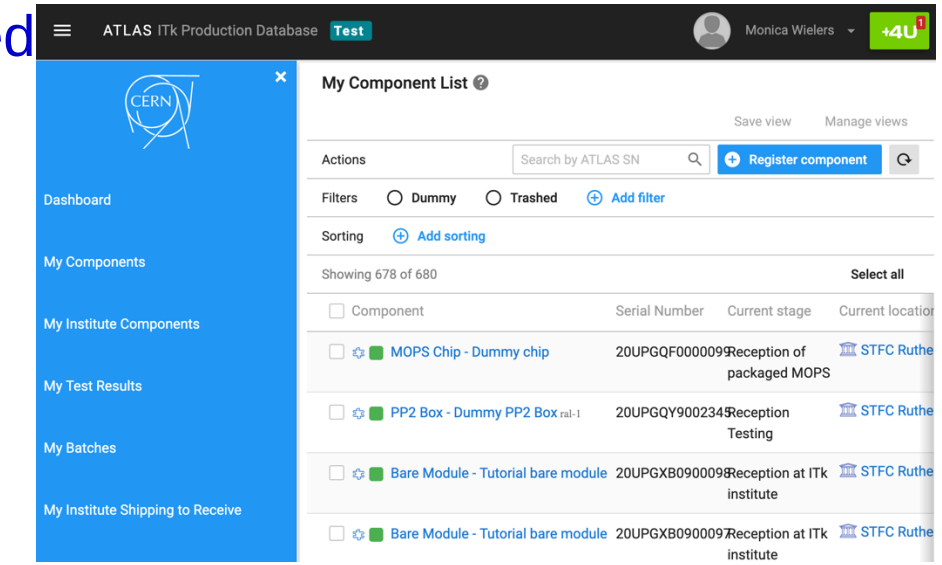


- We now have a database based on MongoDB running in cloud in place

Production Database Component and Test Set-up

- Already >400 component and >2000 test types defined
  - Properties and parameters set-up for everything which needs being recorded

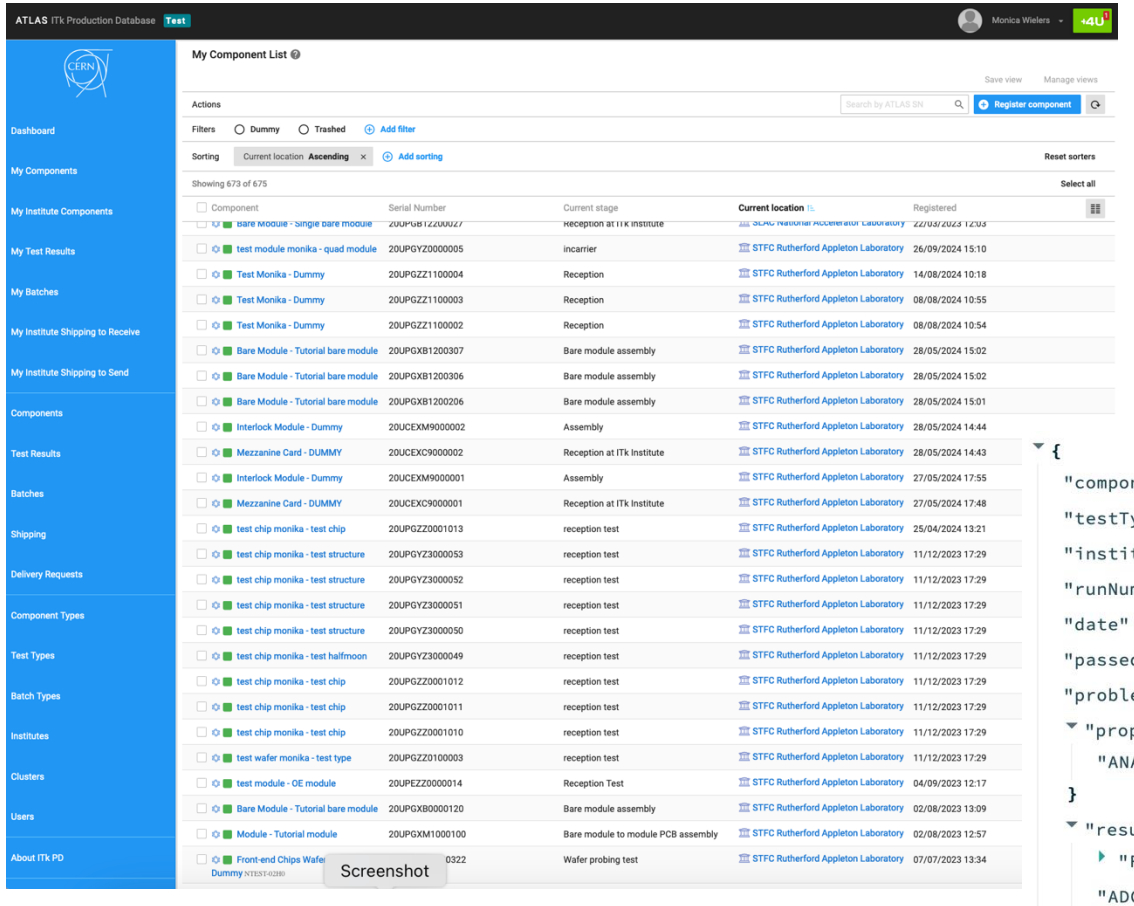
Sub-project	component types	test types
Strips	226	976
Pixels	237	1184



- Possibility to add attachments to components or tests
  - Needed for visual inspection images, raw data for some electrical tests to redo test decisions, configuration files
  - Found out this is more commonly used than expected
- Handled in 2 ways to keep costs for low
  - Usage of binary storage attached to database if file < 64kB
  - Use CERN EOS cloud space for files > 64kB (not integrated as integral part in database)
- Needs careful monitoring and clear rules to ensure appropriate use
  - The typical answer ‘just in case’ is not good enough

Production Database Interactions

User interface



API commands

- Mostly often used in our community
  - Easy to develop scripts
  - Allows development of custom scripts/web apps for data uploads
- Allows
  - Batch upload via csv/excel files or google sheets
  - Machine output can be converted to format needed for DB uploads
  - Sanity checks help avoiding human mistakes
- Uploads done using json format
- ITk community also developed api wrappers (itkdb, dbAccess)
  - Eases script developments

Tools to interact with Production Database

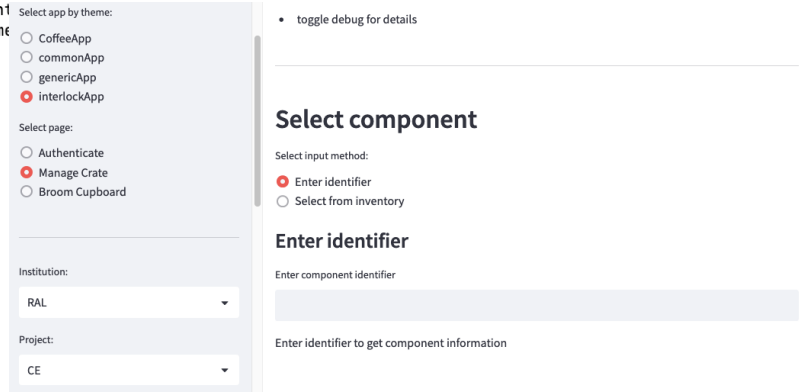
- General purpose tools and tools for dedicated tasks developed by ITk collaboration
  - GUIs, webapps, notebooks and command-line scripts
- Usage depends on subgroup/institutes
  - Important to tailor tools to needs in diverse community (technicians, engineers, physicists)
- Tools distributed via CERN's IT platforms & licences

Python scripts distributed via git

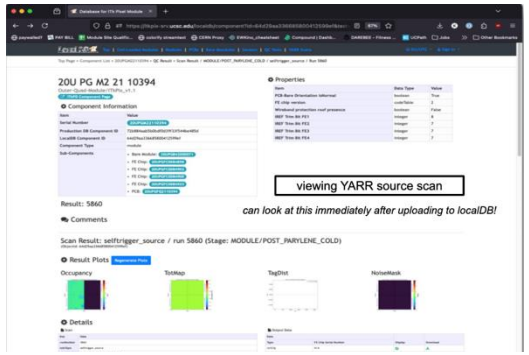
```
monika.wielers@HEPDOCK213 production_database_scripts % python3 registerComponent.py

[INFO]$ Running ITk Production Database component registration interface.
dbAccess:  Getting token.
dbAccess:  Token already exists in shell environment.
[INFO]$ Updating list of institutions.
[PROMPT]$ To always print the available input options for codes, please type 'y/Y' or t
ype 'n/N' to suppress this output:
n
[INFO]$ Use escape codes &PRINT to print the available options, &JSON to print
ent JSON for your component, or &CANCEL to cancel the registration at any time
[PROMPT]$ Enter your institution code:
RAL
[INFO]$ Using code: RAL (STFC Rutherford Appleton Laboratory)
```

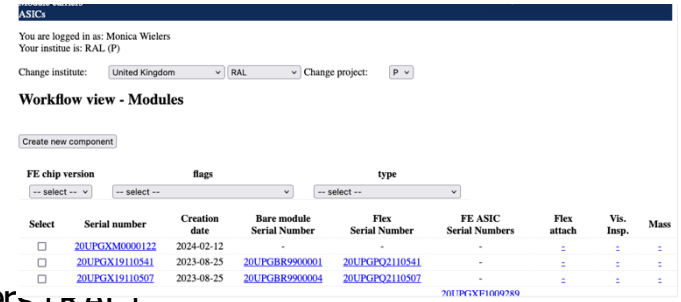
Streamlit (multi-purpose tool) hosted on CERN OpenShift in docker



LocalDB for pixel modules distributed via git



Flask for pixel modules hosted on CERN OpenShift in docker image



Database Usage

- Current information in database
  - Note: production for many parts has not yet started

 # registered components

Project	# components	# institute	# user
Pixel	~170k	~70	~300
Strips	~860k	~70	~450

 # uploaded tests

$\sim 9.8 \cdot 10^6$

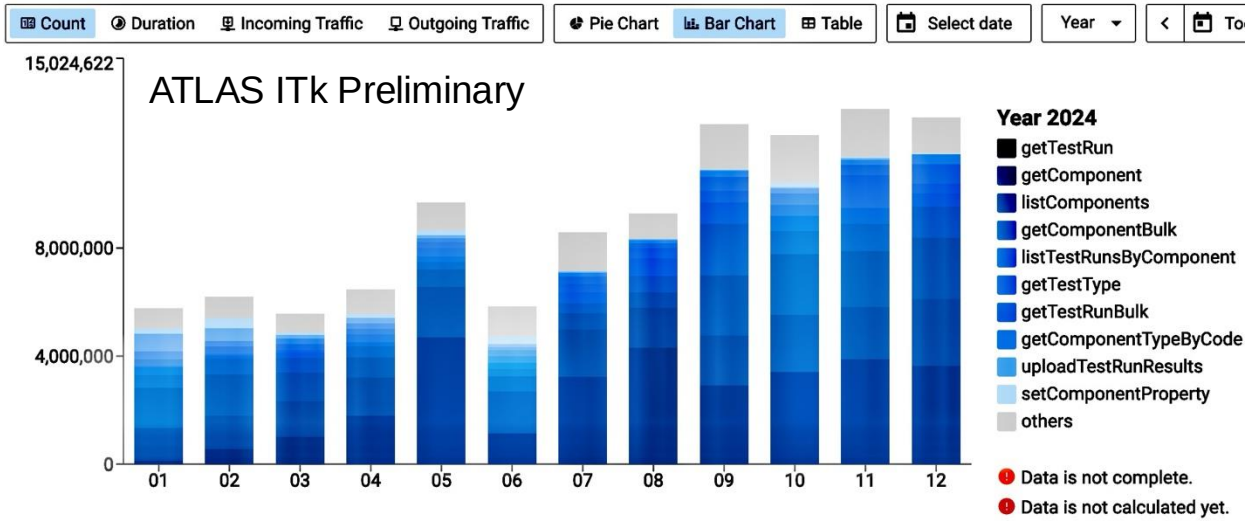
 # shipments

~9000

- Number of requests sent to database
  - Reading out data far outstrips inputting

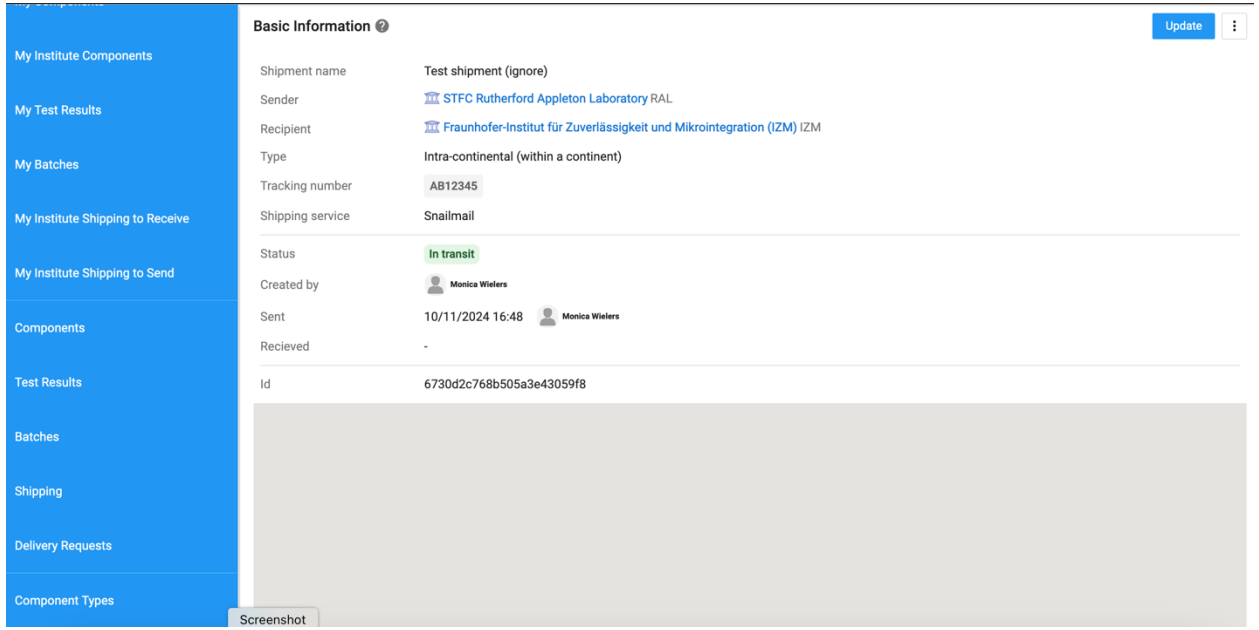
Top 3 commands

getTestRun	39%
getComponent	21%
listComponents	12%



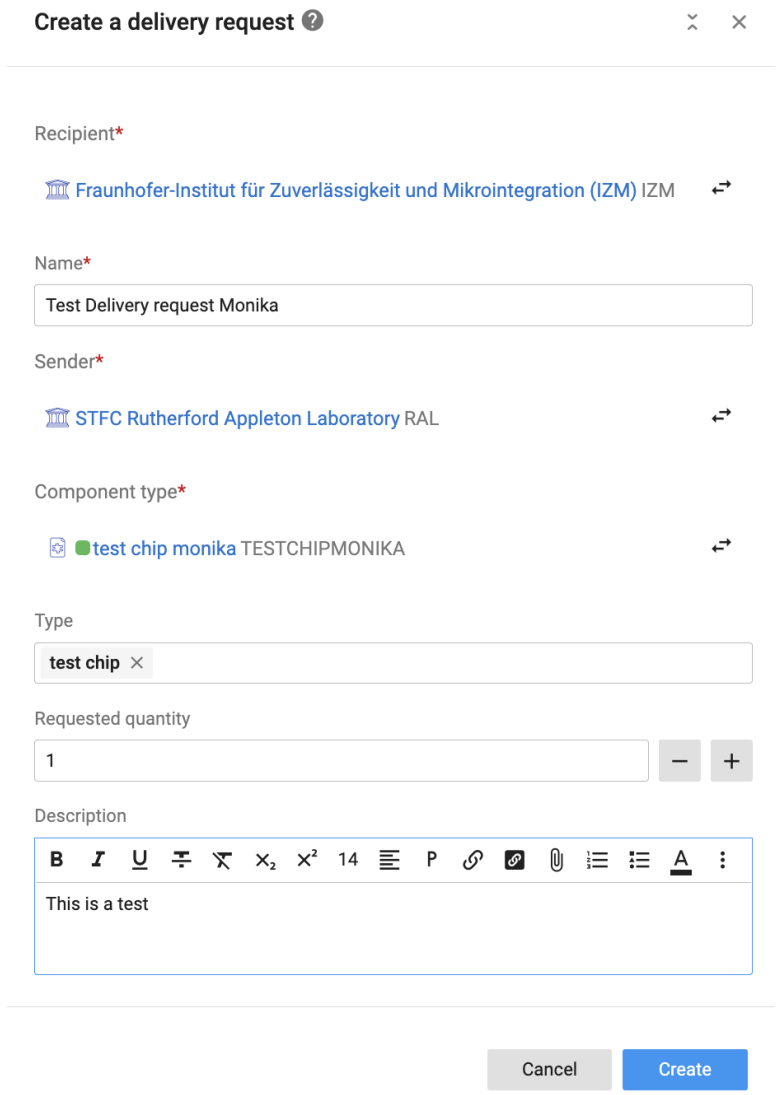
Shipments

- Integral part of database
  - Important feature for production as it allows to track location of parts
    - This is in particular important for parts falling under dual-use export licences
      - Implies all parts need to be shipped back to institute who bought these parts (CERN in our case), can be individual items or larger structures on which these parts were mounted
      - Make sure all parts falling under this have a unique vendor number to ensure traceability
  - When sending a shipment, people at receiving institute are notified by mail
    - This includes the information on shipping service and tracking number
  - When accepting a delivery, user fills out checklist to track if any damage occurred during shipment
  - Upon delivery, sending institute is notified by mail



Delivery Requests

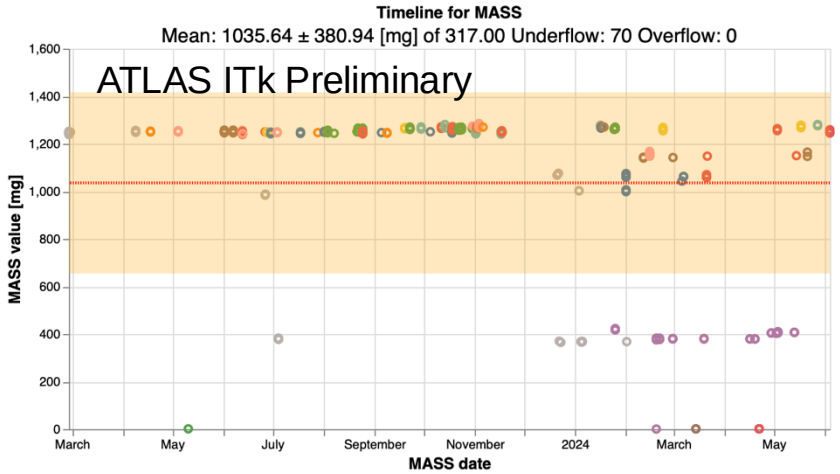
- This is a very useful additional feature
  - Way to ensure all institutes have enough parts in hand
  - Dashboards can show how many parts are available and ready to be shipped to building sites
- Institutes can request parts from other institutes, or are requested by part flow managers
  - These requests are typically discussed in the strips and pixel production group meetings. Ensures parts are shipped in a fair way to all institutes involved
- Approval of delivery requests (including the number of parts which are approved) done by part flow managers
- Once approved, the delivery request is added to a shipment and closed once fulfilled



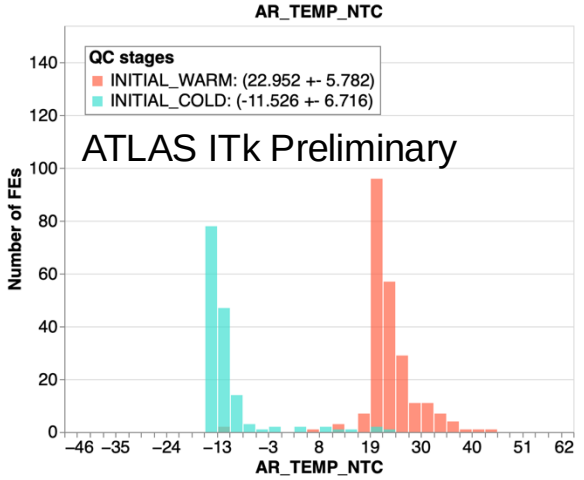
Reporting (done outside database)

- Common tools to create reports
  - Flattening, Visualisation (tables, reports), Distribution (creation of data panels)
- Type of reports
  - On-demand reports
  - Scheduled reports
  - Alerts in case parts do not fulfill specifications
- All reports end up in single entry point (reporting hub)
  - Different level of details needed for sub-project coordinators, production management group, sub-project analysers, institute level
- Reporting being done for
  - Checking data consistency
  - Monitoring of parts location and production rates
  - Monitoring of production quality and yields
- Reporting takes up lots of resources! We will use the weekly backup copy hosted at CERN this year

Timeline of mass measurement



Testing temperature in cold/warm



Data Mining

- Important to look at data in case issues during production/operations arise
  - Try to find correlations in data
    - In which area of the component did the problem occur
    - Does it affect one batch only
    - Does it affect parts from one vendor only
    - Which test results does it affect
    - Does it affect certain production version
    - Does it happen at the same stage in the production flow
    - ….
- Successfully used database for these purposes in strips and pixel sub-projects
  - Database data-mining essential to understand issues!

What I like/like less about the database

- DB ensures all the component information and QC/QA test results are in one central place
- Accessible via API commands for script/webapp development
- Front-end and back-end are well separated
  - Back-end maintained by vendor
  - Front-end by ITk community
- Shipment and delivery requests part of the database features
  - Allows tracking of dual-use export licenses
- As DB comes from vendor API commands are their proprietary software and cannot be used on the CERN-based backup
- DB hosted by vendor, so no close integration with eos space for attachments at CERN possible
- We did not provide some recommended/example tools to interact with the database in time
  - Now everyone uses his/her solution, which makes it more complicated that the test pass/fail decisions are done in the same way for all component types with a simple interface to access to the analysis code

Conclusions

- ATLAS ITk production database is a fully functional distributed data management system in use for the last 4 years by ~100 institutes
- Database heavily used for registration and test uploads
  - ~10^6 components registered and nearly 10×10^6 test results in database
- Vital tool for monitoring production to ensure detector build on time and to specifications
- Database will be operational for the remaining years of construction and during HL-LHC operation
  - Long-term maintenance of database and the front-end tools used for reporting needed

Backup

Production Database Technical Setup

- The ITk Production Database based on mongoDB
- Two parts
  - User interface (front-end)
  - Server side (back-end)
- Based on the uuApp Framework developed by vendor
  - Framework is based primary on open-source technologies
  - Fully designed as a cloud-based application
    - Operated in uuCloud, which is powered by MS Azure
- Terminology
  - A server-side functionality is called uuCmd and is represented by an endpoint
  - The API can be called via HTTP requests (GET or POST method)
- Front End (scripts/web apps) is developed by the wider ITk community