iDDS
(intelligent Data Delivery Service)

Wen Guan, Tadashi Maeno, Gancho Dimitrov
Brian Bockelman, Torre Wenaus
On behalf of iDDS project

March 11, 2020
3rd Rucio Community Workshop

Introduction

- **An intelligent service to transform and deliver needed data to consumers.**
  - Not a storage, WFMS or DDMs
  - Delegation of many functions to backend systems, such as WFMS and DDMs, and infrastructure, such as cache and network.
- Requirements
  - Flexibility to experiment agnostic
  - Flexibility to support many use-cases and backend systems
  - Easy and cheap to deploy
- Key functions
  - Transformation
  - Delivery
  - Orchestration

Why new service

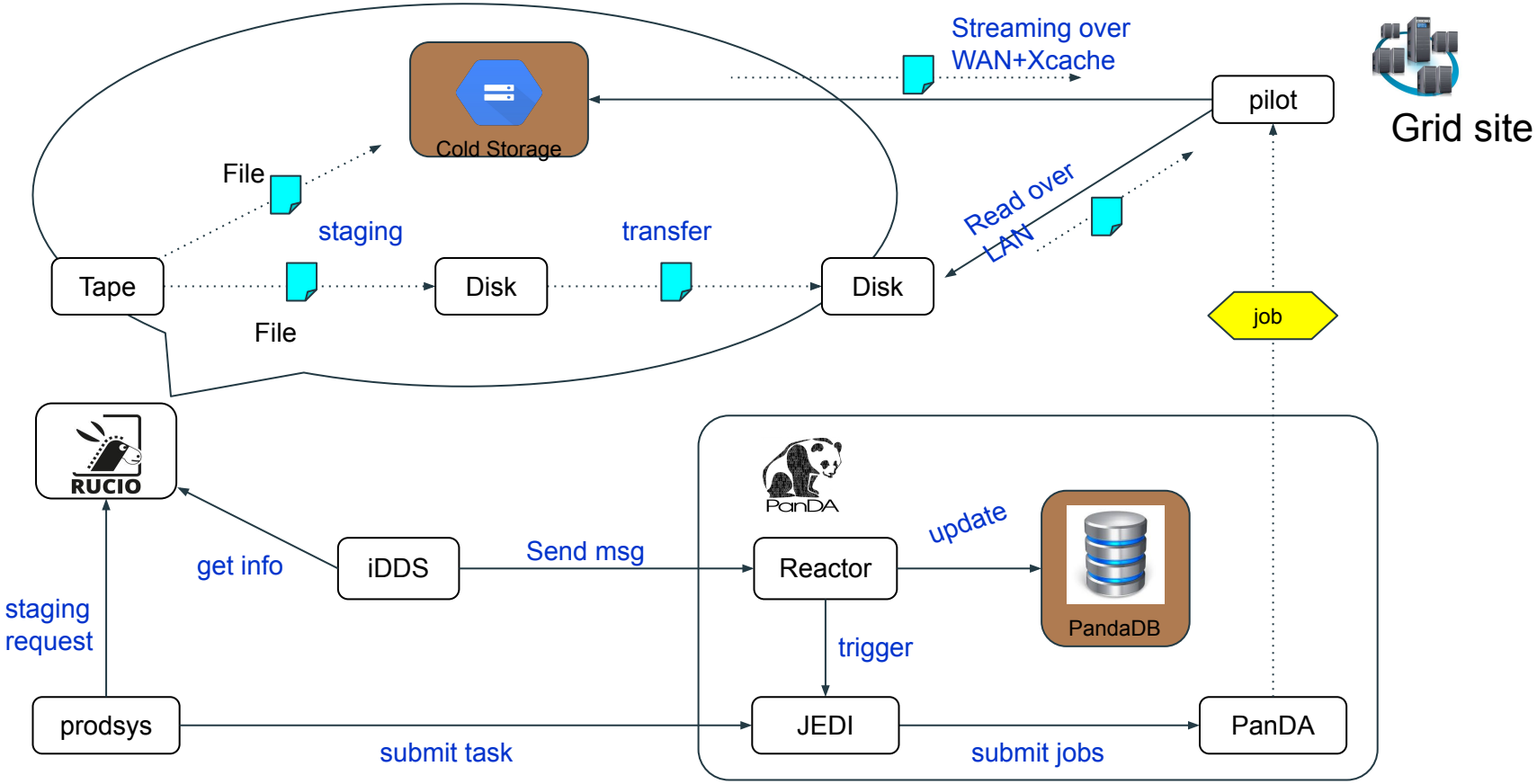
- **Orchestration of WFMS and DDMS is crucially important for optimal usage of limited resources**
  - **Resources = CPU, disk and tape storages, network, and manpower**
  - **Delivery of necessary data to compute resources just in time**
  - **Rapid deletion of data as soon as they are processed**
- **Needs for orchestration have been "discovered" while accumulating operational experiences, and functions have been added to existing services without generalization**
  - **Examples in ATLAS**
    - **Sub data block replication → PanDA**
    - **Dynamic data placement → PanDA and Rucio**
    - **Tape carousel → prodsys**
    - **Fine grained processing → JEDI**
  - **Ambiguous and overwrapped service boundaries**
  - **Lack of reusability in each function for another usecase**
- **New use cases**
  - **Fine grained tape carousel**
  - **Active Learning**
  - **Hyperparameter tuning**
  - **Event Streaming Service**
  - **On-demand production of analysis format data**

iDDS

- A general service to transform and deliver needed data to consumers.
  - Orchestration of WFMS and DDMS with generalized workflows
    - Experiment agnostic based on the generalization
    - Extraction and abstraction of functions for orchestration
    - Maintainability and extensibility with plugin architecture
  - Flexibility to support new workflows
    - Fine grained tape carousel
    - Active Learning
    - Hyperparameter tuning
    - Fine grained data transformation
    - Remote data transformation/reduction
    - On-demand production of analysis format data

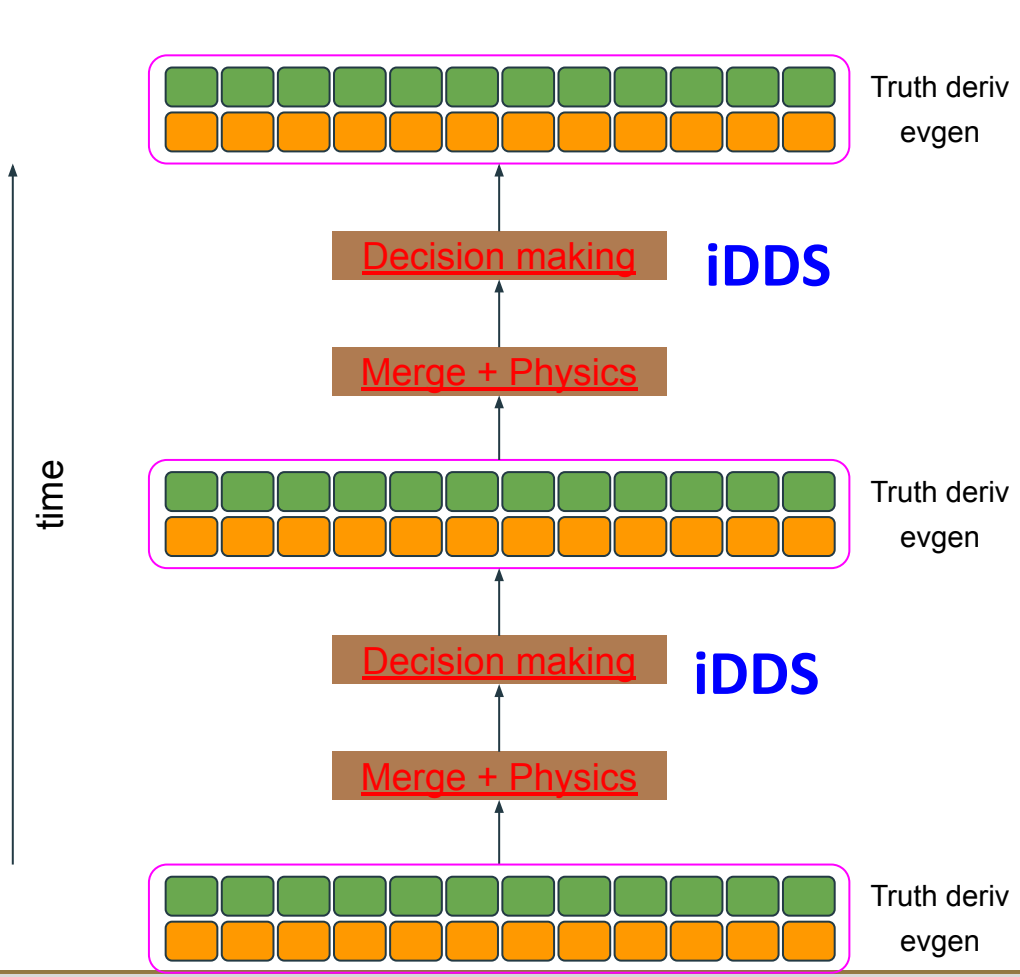
Fine grained Data Carousel (implemented)

- **Talks to rucio to collect and digest file information, and lets JEDI/PanDA process only prestaged files with proper granualities and grouping**
- **Decides access protocol depending on data location etc in the future**
  - E.g. direct reading for google cold storage to avoid redundant egress due to job reassignment



Decision Making for Active Learning 1/2

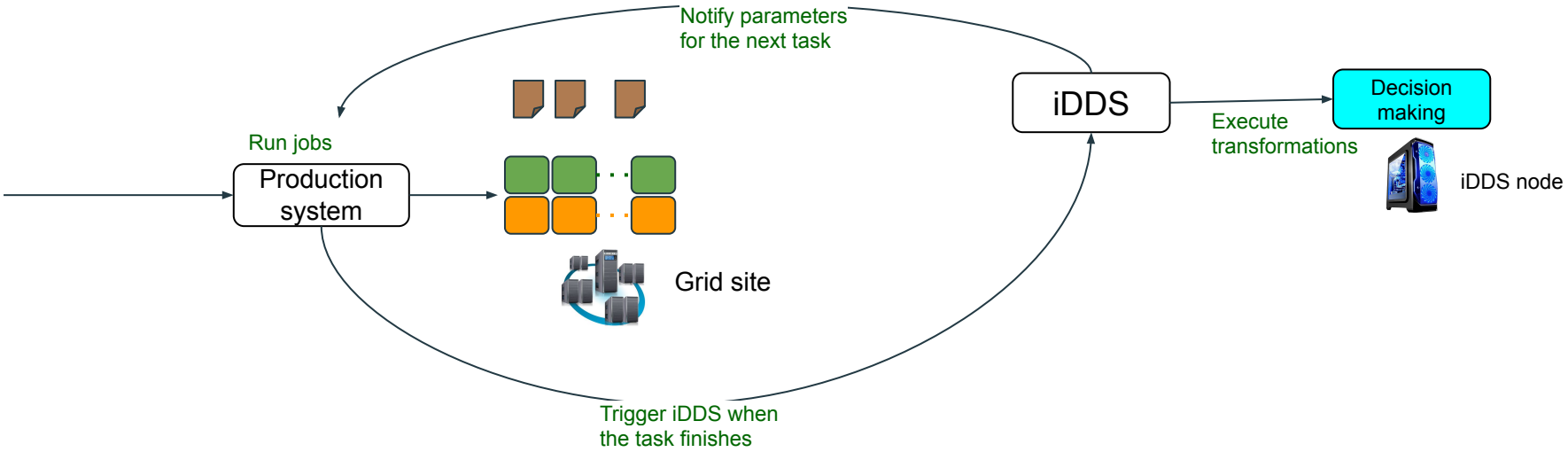
- Active learning
  - Running tasks on top of results of old tasks
  - Decision making to generate new tasks from old results
- Workflow with grid entities
  - Production system processes the normal task
  - iDDS runs the Decision Making parts(with/without merge parts)



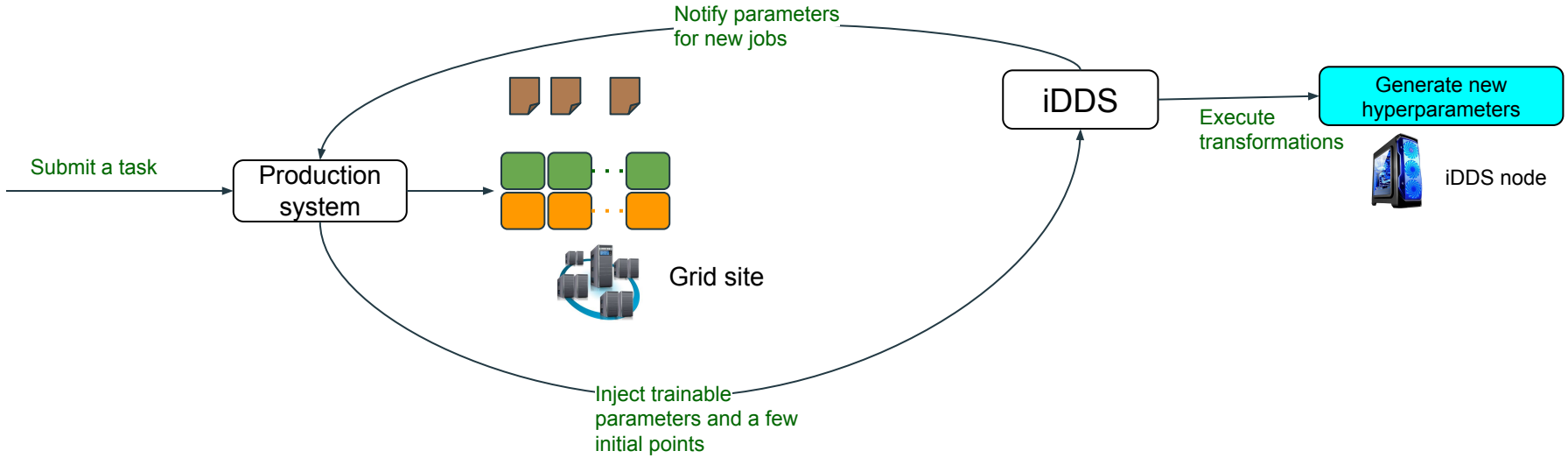
- Each job runs evgen and derivation sequentially
- Each task generates many multi-step jobs
- Once the first task is finished output are merged and some values are calculated
- Parameters for the next task are decided using the values
- Better to get rid of latency in the intermediate steps (merge+physics and decision making)

Decision Making for Active Learning 2/2

- **What should run intermediate steps?**
  - **Task/Job : the first task → merge/physics task → decision making task → the next task → …**
    - Possible to use task chaining
      - Base on the finished task, iDDS triggers to generate new tasks
    - Tasks require a few CPU time
      - Decision making requires a few CPU minutes
      - Merge/phys is can be an option if it's not CPU intensive and not disk intensive
  - **External service for quick turnaround → iDDS**
    - Production system triggers iDDS to run the decision making job when a task finishes
    - iDDS reads the results of the decision making job and notifies production system to generate new tasks
    - transformation backends
      - Users provided codes as a decision making transformation backend
      - Builtin function to allow users to execute arbitrary transformations
  - **Dynamically generate new tasks**



Hyperparameter tuning



- **Workflow**
  - The User submits a hyperparamter tuning task
    - With trainable hyperparameters and ranges
    - With stopping parameters
    - With a few initial points (optional)
  - Production injects the trainable hyperparameters and ranges to iDDS, and initial points if available
  - iDDS generates a batch of initial points if not defined.
  - iDDS notifies the batch of hyperparameters to to Production system
  - Production system runs jobs with received hyperparameters and notify results to iDDS
  - iDDS generates new batch of hyperparameters for jobs in the task
- **Dynamically generate new jobs in the same task**
- (https://docs.google.com/presentation/ d/1fGw_p9XMHhbzH7GV0MavoE75OEk4tVdH0F-4YMZvwM/edit?usp=sharing)

Other usecases

On-demand production

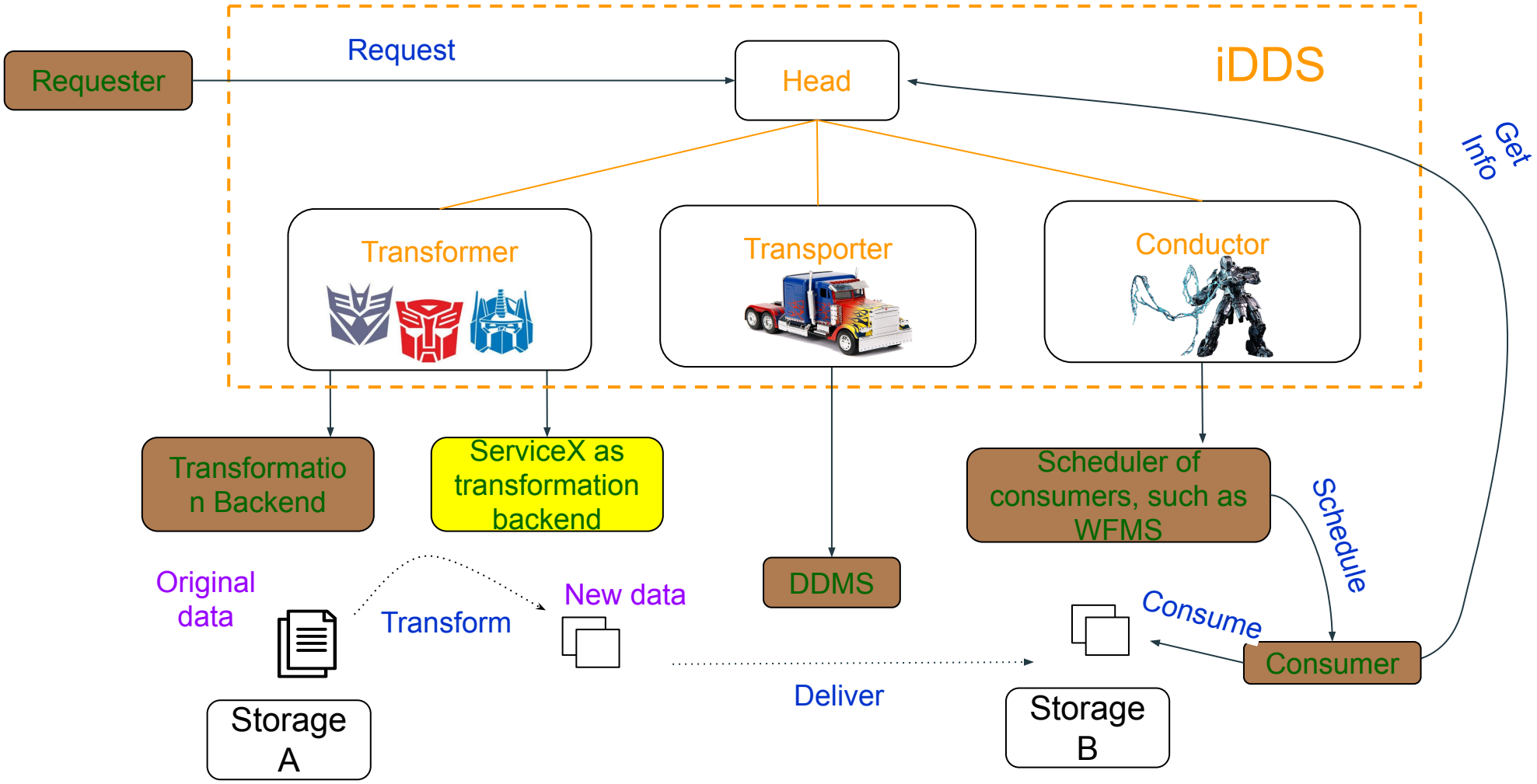
- E.g. to produce unpopular DAOD after users submit analysis tasks
- Trade-off between CPU and disk
- Advantages
  - Less disk usage and No unused datasets
- Disadvantages
  - Latency before analysis tasks actually get started
  - Issue with reproducibility
    - Re-created DAOD are not completely identical although they should be consistent to original files within numerical fluctuations
- ↔ Proactive production
  - No latency, more disk usage, production of unused datasets
  - Mainstream files like DAOD_PHYS(LITE)
- Ability to allow physics working groups to customize contents
  - TBD since not in ATLAS computing model

Remote reduction/transformation

- Centralization of data conversion for subsequent processing
  - Skimming, slimming, re-formatting, …
- Would work well with data lake/ocean
- Reduction of data transfer over WAN, removal of data conversion step from iterative processing, geographical distribution of subsequent processing

Integration with ServiceX

- **Plugin structure in Transformer agent**
  - Selection of plugin depending on use cases
- **A plugin to use ServiceX as a transformation backend**



Current Status

- **Main architecture --ready**
  - iDDS database, core, REST API
  - Plugins
  - Agents
  - Watchdogs
  - Documentation: to add contents to https://idds.cern.ch
- **Use cases**
  - Fine-grained data carousel -- ready, integration tests with panda
  - Decision making for active learning -- developing
  - Hyperparameter tuning -- next
  - Other usecases in 2020
- Monitors
  - working with monitoring team
  - Prototypes are ready within panda monitor and grafana.