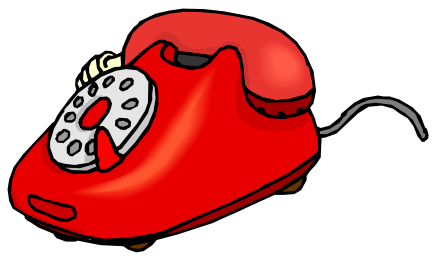
DIAL
Distributed Interactive Analysis of Large datasets

CHEP 2003

Data Analysis Environment and Visualization

David Adams
BNL
March 25, 2003



Contents

Goals of DIAL

What is DIAL?

Design

- Applications
- Schedulers
- Datasets

Status

Development plans

GRID requirements

Goals of DIAL

1. Demonstrate the feasibility of interactive analysis of large datasets
   - Large means too big for interactive analysis on a single CPU
2. Set requirements for GRID services
   - Datasets, schedulers, jobs, resource discovery, authentication, allocation, ...
3. Provide ATLAS with analysis tool
   - For current and upcoming data challenges

What is DIAL?

Distributed

- Data and processing

Interactive

- Prompt response (seconds rather than hours)

Analysis of

- Fill histograms, select events, …

Large datasets

- Any event data (not just ntuples or tag)

What is DIAL? (cont)

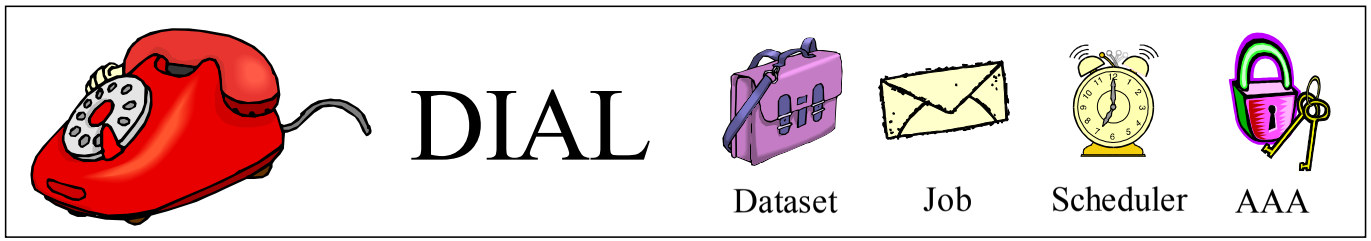
DIAL provides a connection between

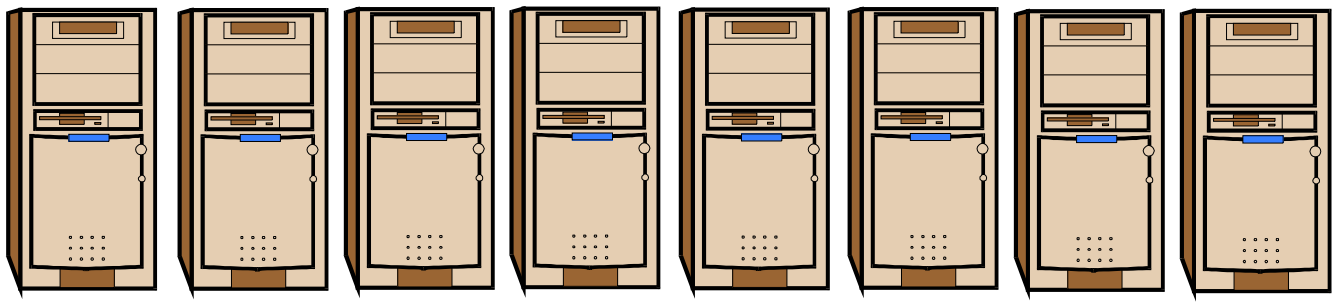
- Interactive analysis framework
  - Fitting, presentation graphics, …
  - E.g. ROOT
- and Data processing application
  - E.g. athena for ATLAS
  - Natural for the data of interest

DIAL distributes processing

- Among sites, farms, nodes
- To provide user with desired response time

Interactive analysis
e.g. ROOT, JAS, ...



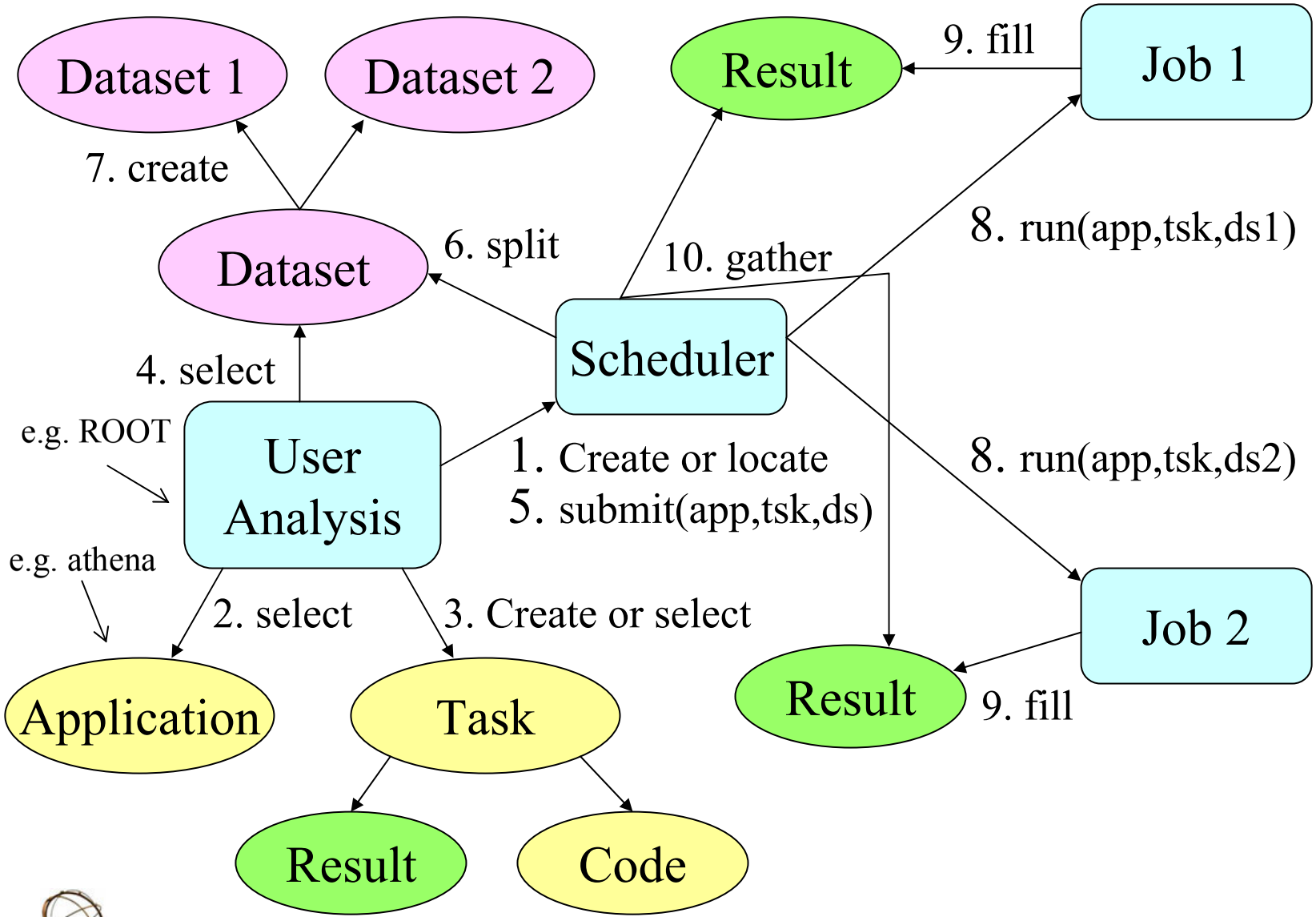


Distributed processing running data-specific application

Design

DIAL has the following components

- Dataset describing the data of interest
  - Organized into events
- Application
  - Event loop providing access to the data
- Task
  - Result to fill for each event
  - Code process each event
- Scheduler
  - Distributes processing and combines results

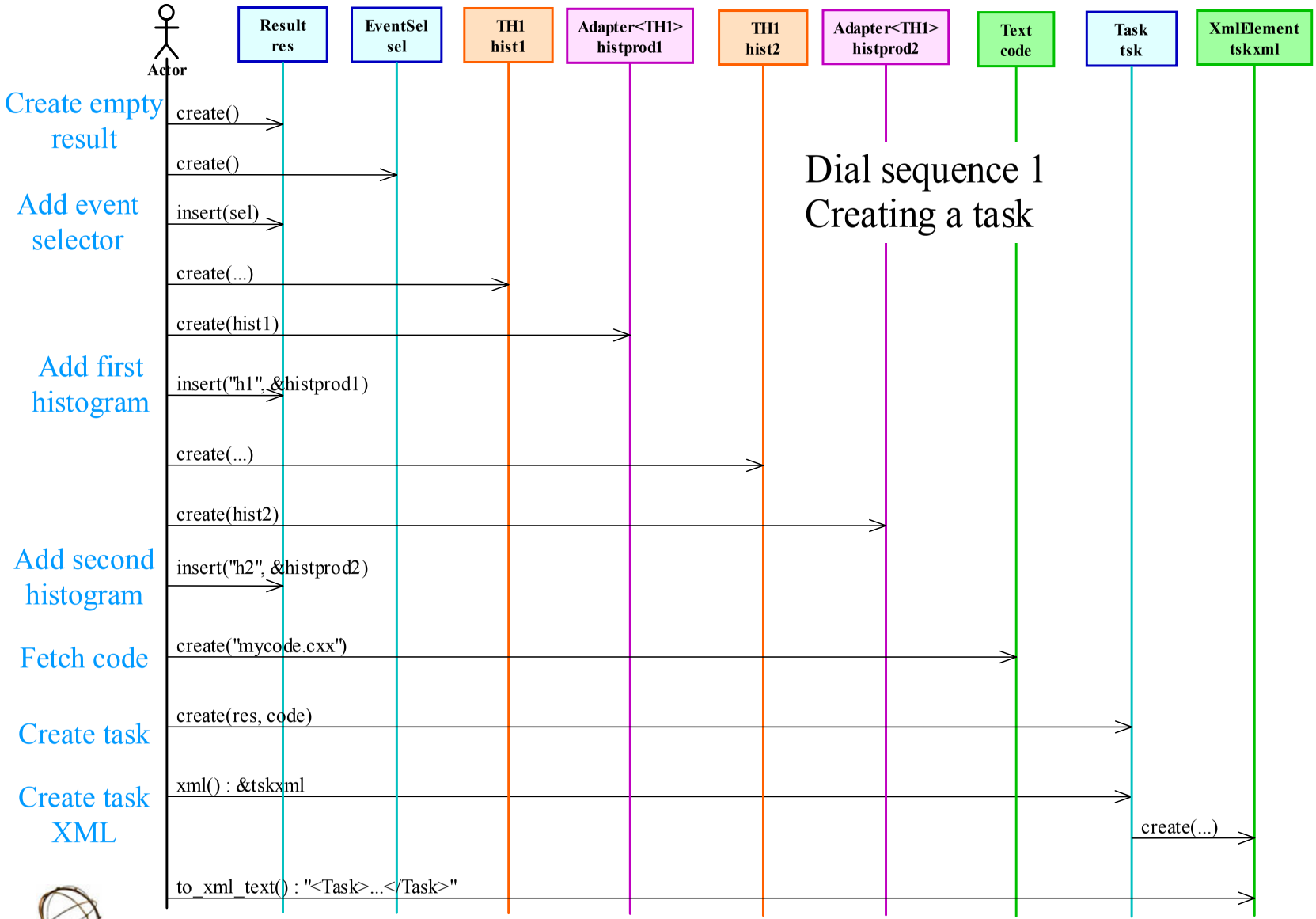

Dataset 1
Dataset 2
Result
9. fill
Job 1
7. create
Dataset
6. split
10. gather
8. run(app,tsk,ds1)
Scheduler
4. select
e.g. ROOT
User Analysis
1. Create or locate
5. submit(app,tsk,ds)
8. run(app,tsk,ds2)
e.g. athena
2. select
3. Create or select
Job 2
Application
Task
Result
9. fill
Result
Code

Design (cont)

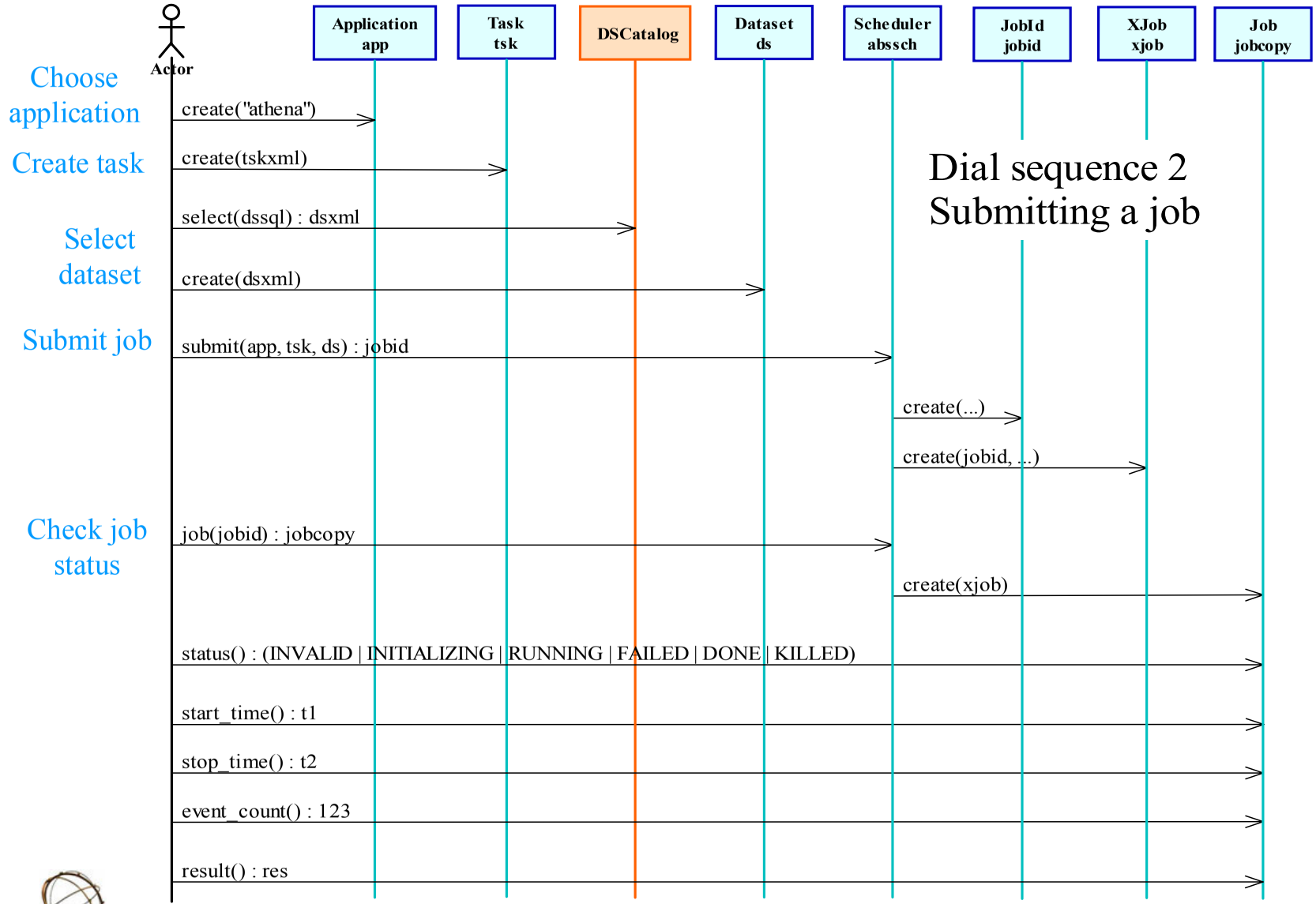
Sequence diagrams follow

1. User creates a task made up of
   - Event selection
   - Two histograms
   - Code to fill these
2. User submits a job (application, task and dataset) to an existing scheduler
3. Grid scheduler uses site schedulers to process a job

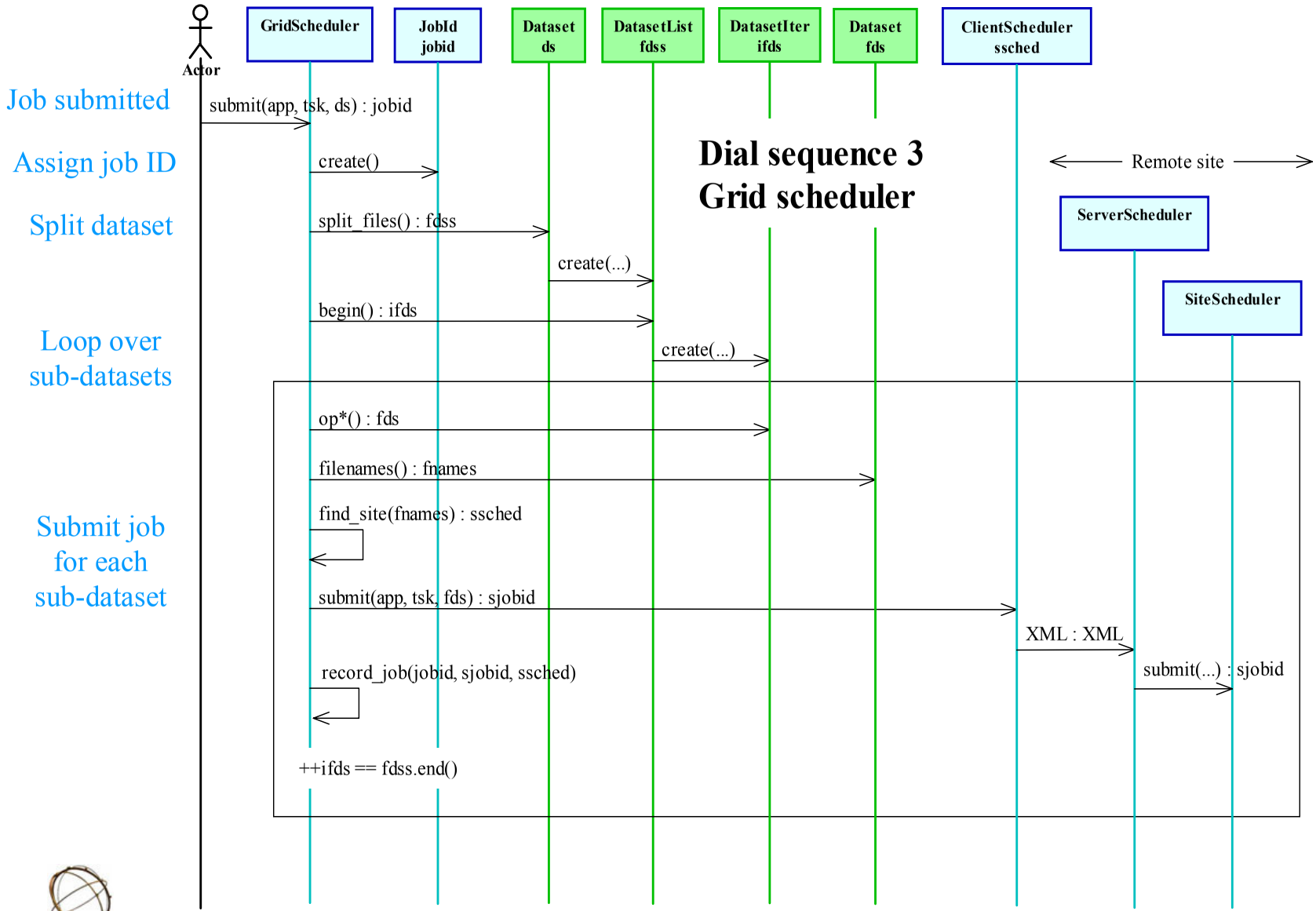
Dial sequence 1
Creating a task



Dial sequence 2
Submitting a job



Dial sequence 3
Grid scheduler



Applications

Current application specification is

- Name
  - E.g. athena
- Version
  - E.g. 6.10.01
- List of shared libraries
  - E.g. libRawData, libInnerDetectorReco

Applications (cont)

Each DIAL compute node provides an application description database

- File-based
  - Location specified by environmental variable
- Indexed by application name and version
- Application description includes
  - Location of executable
  - Run time environment (shared lib path, …)
  - Command to build shared library from task code
- Defined by ChildScheduler
  - Different scheduler could change conventions

Schedulers

A DIAL scheduler provides means to

- Submit a job
- Terminate a job
- Monitor a job
  - Status
  - Events processed
  - Partial results
- Verify availability of an application
- Install and verify the presence of a task for a given application

Schedulers (cont)

Schedulers form a hierarchy

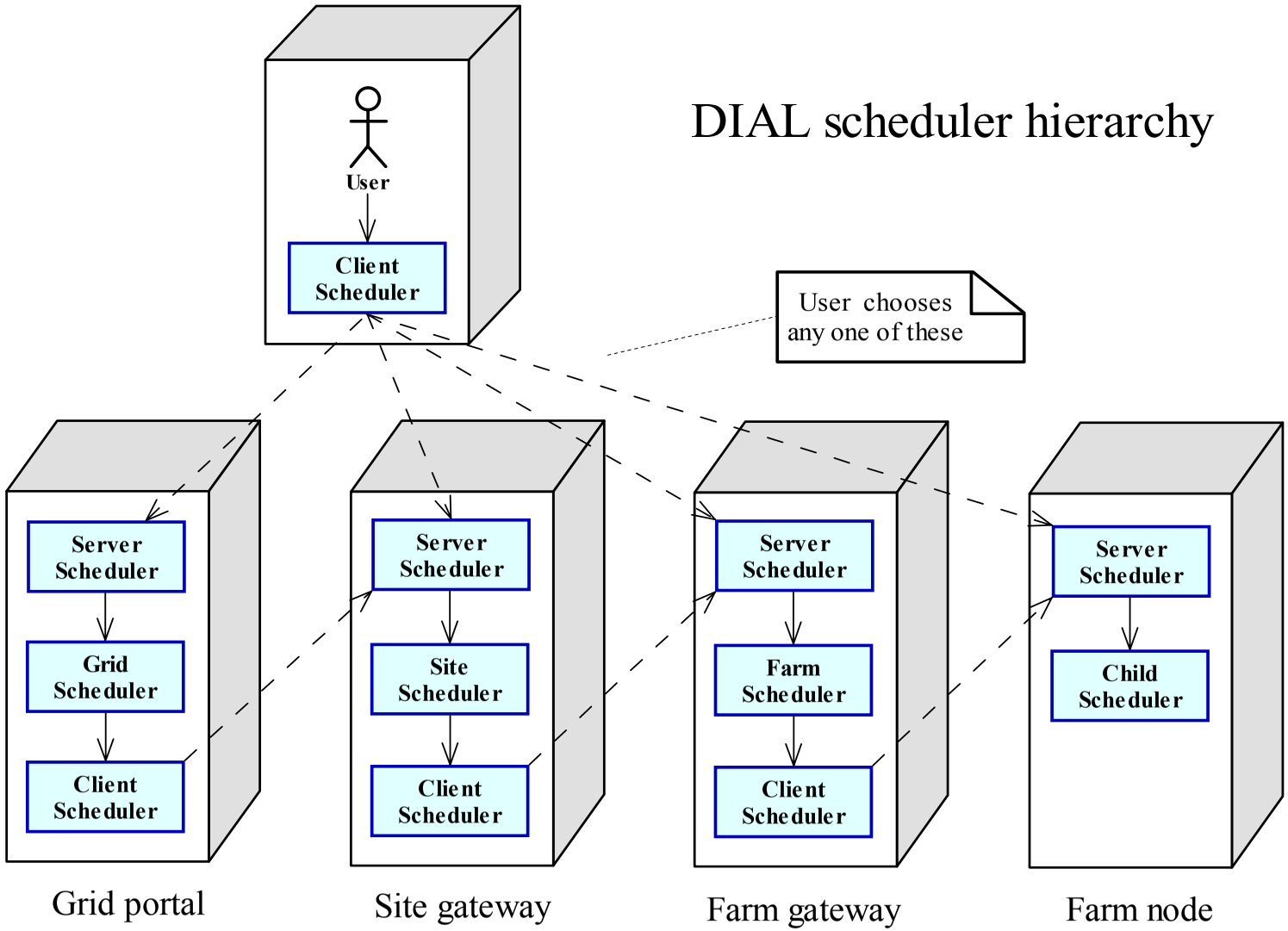
- Corresponding to that of compute nodes
  - Grid, site, farm, node
- Each scheduler splits job into sub-jobs and distributes these over lower-level schedulers
- Lowest level ChildScheduler starts processes to carry out the sub-jobs
- Scheduler concatenates results for its sub-jobs
- User may enter the hierarchy at any level

Schedulers (cont)

Schedulers communicate using client-server

- Between processes, nodes, sites
- User constructs a client scheduler specifying
  - Remote node
  - Name for remote scheduler
- Server process on remote machines
  - Starts schedulers and assigns them names
  - Passes requests from clients to the named scheduler
- Not yet implemented
  - Communication protocols not established

DIAL scheduler hierarchy

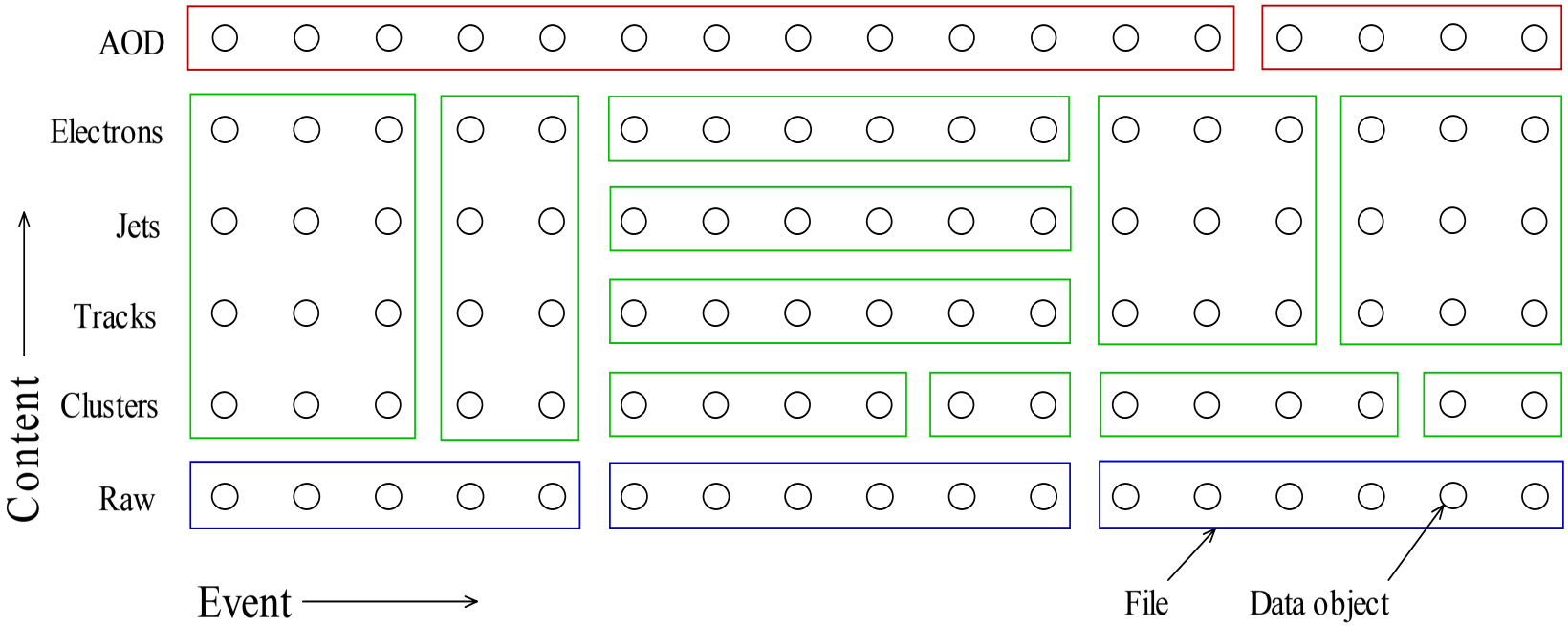


Datasets

Datasets specify event data to be processed

Datasets provide the following

- List of event identifiers
- Content
  - E.g. raw data, refit tracks, cone=0.3 jets, …
- Means to locate the data
  - List of of logical files where data can be found
  - Mapping from event ID and content to a file and a the location in that file where the data may be found
  - Example follows

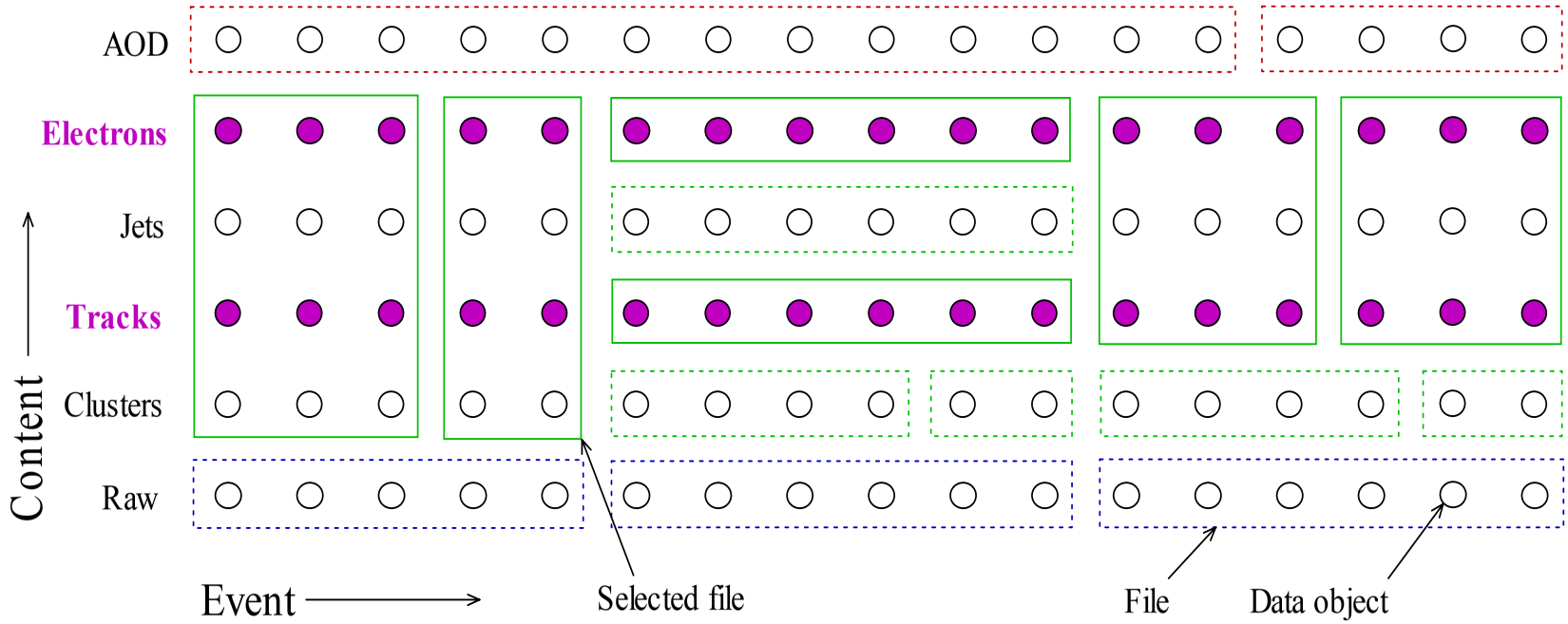


Example dataset with mapping to files

Datasets (cont)

User may specify content of interest

- Dataset plus this content restriction is another dataset
- Event data for the new dataset located in a subset of the files required for the original
- Only this subset required for processing

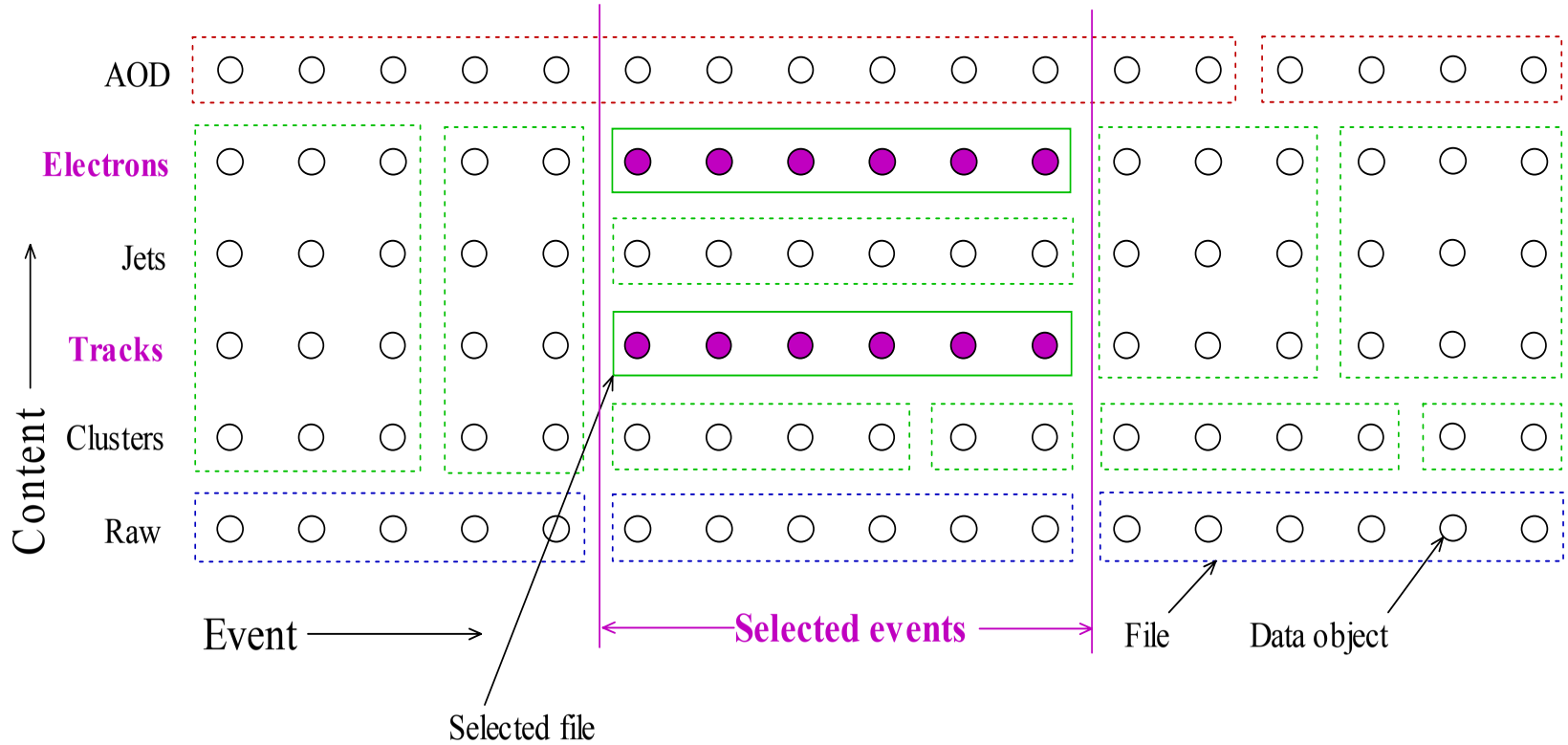


Example dataset with content selection

Datasets (cont)

Distributed analysis requires means to divide a dataset into sub-datasets

- Sub-dataset is a dataset
- Do not split data from any one event
- Split along file boundaries
  - Jobs can be assigned where files are already present
  - Split most likely done at grid level
- May assign different events from one file to different jobs to speed processing
  - Split likely done at farm level



Example sub-dataset with content selection

Status

All DIAL components in place

- http://www.usatlas.bnl.gov/~dladams/dial
- But scheduler is very simple
  - Only local ChildScheduler is implemented
  - Grid, site, farm and client-server schedulers not yet implemented

Datasets implemented as a separate system

- http://www.usatlas.bnl.gov/~dladams/dataset
- Only concrete dataset is ATLAS AthenaRoot
  - Holds Monte Carlo generator information

Status (cont)

DIAL and dataset classes imported to ROOT

- ROOT can be used as user interface
  - All DIAL and dataset classes and methods available at command prompt
  - DIAL and dataset libraries must be loaded
- Import done with ACLiC
- Only preliminary testing done
- Need to add adapter for TH1 and any other classes of interest

DIAL status (cont)

No application integrated to process jobs

- Except test program dialproc can be used to count events
- In ATLAS natural thing is to define a DIAL algorithm to run in athena
  - However ATLAS is not yet able to persist reconstructed data
- Perhaps a ROOT backend to process ntuples?
  - Or is this better handled with PROOF?
  - Or use PROOF to implement a farm scheduler?

Development plans

(Items in red required for useful ATLAS tool)

Schedulers

- Add client-server schedulers
- Farm scheduler
  - Allows large-scale test
- Site and grid schedulers
  - GRID integration
  - Interact with dataset, file and replica catalogs
  - Authentication and authorization

Development plans (cont)

Datasets

- Interface to ATLAS POOL event collections
  - expected in summer
- ROOT ntuples ??

Applications

- Athena for ATLAS
- ROOT ??

Analysis environment

- Import classes into LCG/SEAL? (Python)
- JAS? (java binding?)

GRID requirements

Identify components and services that can be shared with

- Other distributed interactive analysis projects
  - PROOF, JAS, …
- Distributed batch projects
  - Production
  - Analysis
- Non-HEP event-oriented problems
  - Data organized into a collection of "events" that are each processed in the same way

GRID requirements (cont)

Candidates for shared components include

- Dataset
  - Events
  - Content
  - File mapping
  - Splitting
- Job
  - Specification (application, task, response time)
  - Splitting
  - Merging results
  - Monitoring

GRID requirements (cont)

- Scheduler
  - Available applications and tasks
  - Job submission
  - Job status including partial results
- Application
  - Specification
  - Installation
- Authentication and authorization
- Resource location and allocation
  - Data, processing and matching

GRID requirements (cont)

Difference with batch processing is ***latency***

- Interactive system provides means for user to specify maximum acceptable response time
- All actions must take place within this time
  - Locate data and resources
  - Splitting and matchmaking
  - Job submission
  - Gathering of results
- Longer latency for first pass over a dataset
  - Record state for later passes
  - Still must be able to adjust to changing conditions

Grid requirements (cont)

Avoid sharp division between interactive and batch resources

- Share implies more available resources for both
- Interactive use varies significantly
  - Time of day
  - Time to the next conference
  - Discovery of interesting events
- Interactive request must be able to preempt long-running batch jobs
  - But allocation determined by sites, experiments, …