Exercise: Putting it all together

- Peter collected before the workshop several topics which are of interest to participants
- We compiled these into a list of suitable projects → following pages. Please pick one topic.
- The goal of this exercise is to obtain an almost complete UML description of the project.
- You are free to add features to or to choose your own project, if none on the list interests you.
- Please work in groups of 3-6 people.

Project 1: JetFinder

- Complete the JetFinder project of the Exercise to DesignPatterns II
- (see handout of previous exercise for further instructions)

Project 2: Job submission controller

- The user schedules the submission of job scripts through the controller. The controller submitts the jobs at the specified time to a computer cluster with a system call to the linux tool qsub.
- The controller tests all jobs for successful completion and restarts jobs that failed.
- Allows nice display of status via regular update of a status text file or via a gui.
- Contoller might initiate automated extraction of calibration constants for job outputs

Project 3: HistogramHandler/Manager

- Write a HistogramManager that extracts and displays systematic errors for a given histogram.
- Input:
  - Histogram with the final result (result plot)
  - Several other histograms that represent systematic variations of the result are also provided.
  - For each variation a recipe is provided on how to extract the error
- Output:
  - The result plot in which the statistical AND the total systematic error are displayed as error bars
- Extraction is done in an automated way using a list of recipes
- Additional features:
  - Make project extendable to several result histograms,
  - Allow for a correlation of systematic errors in the calculation of total syst. Error
  - Think about how to pass recipe to Histogram handler (pure text is not recommended)

Example Input

Histo	comment
1	Result plot
2	variation of cut1 by -1 epsilon
3	variation of cut1 by +1 epsilon
4	MCGenerator A used for efficiency
5	MCGenerator B used for efficiency
6	MCGenerator C used for efficiency

Recipe
Histo 1 is Result plot
Take **max.** deviation of histos 2 and 3 as first syst. error 1
Take **average** deviation of 4,5,6 as second error

Project 4: Cutflow manager

- Cutflow manager handles a cut selection defined in a series of ROOT Tcut objects
- The cut selection is applied to one or several datasets which are provided in the form of trees. Each tree has a luminosity weight.
- The number of event after each cut, the absolute and relative efficiency are calculated by the handler
- Possibility of Data/MC comparisons
- Output in a nice table

Project 5: Particle Creator from ntuple branches

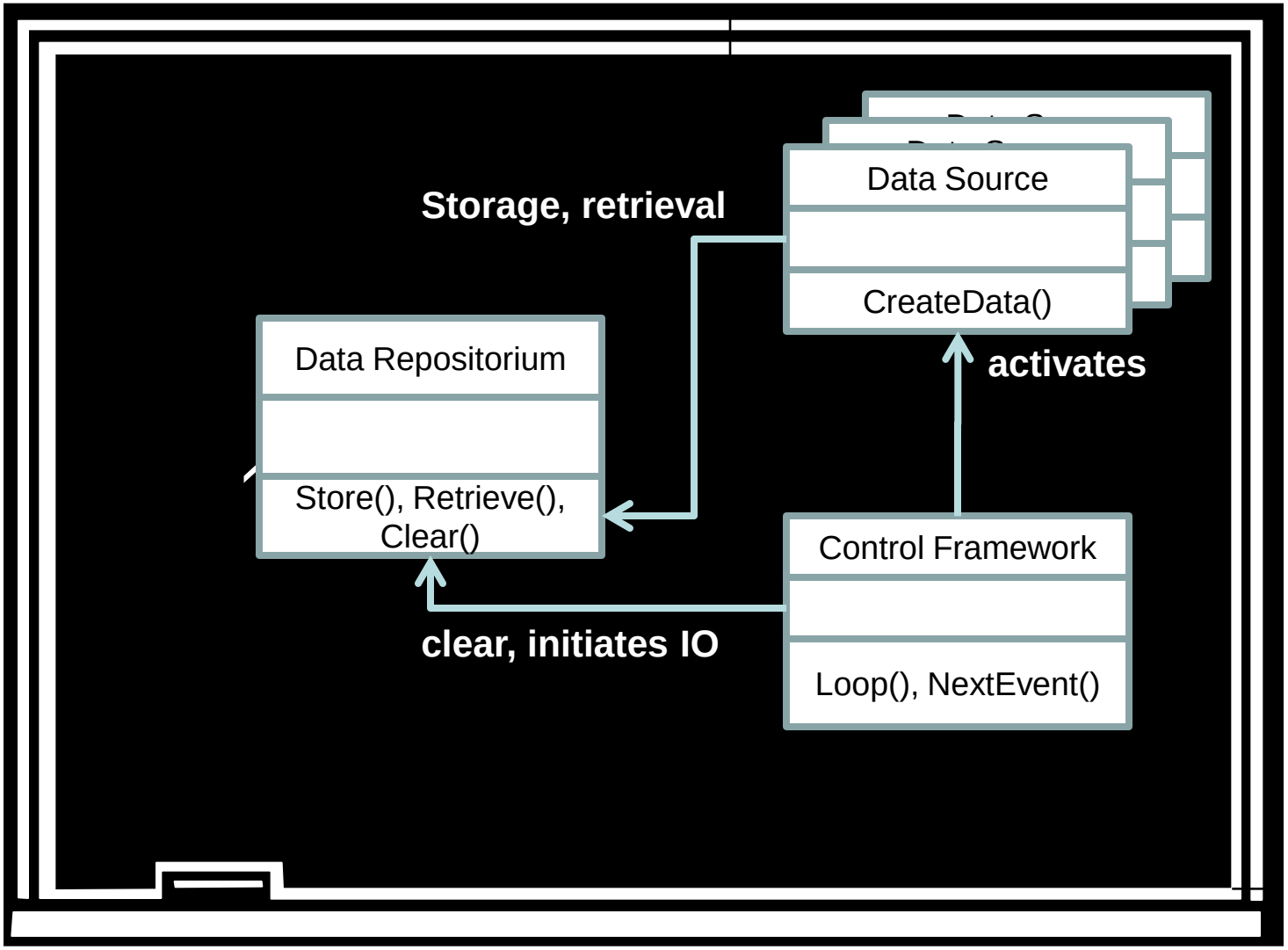
- The particle creator reads entries from a „flat“ tree and creates Particle objects from it. The creator stores all created particles in a transient data storage that follows the blackboard design pattern
- When the particle creator is initialized it reads the tree branches and either automatically or upon request by users marks tree entries for particle creation.
- ParticleCreator has ReadEvent function to read and create patrticles and to store them in the transient data storage
- Example usage

  ParticleCreator::ReadEvent(int ievt)
  - Reads tree
  - Creates vector of particles electrons
  - Stores this vector under Key “List_elec“ in transient data storage
  - Creates vector of particles jets
  - Stores this vector under Key “List_jet“ in transient data storage

Tree content	
type	**varname**
int	Nelec
Float array	elec_px
Float array	elec_py
Float array	elec_pz
Float array	elec_e
int	Njet
Float array	jet_px
Float array	jet_py
Float array	jet_pz
Float array	jet_e

Particle Creator from ntuple branches

Blackboard Pattern



Particle Creator from ntuple branches

Blackboard pattern in HEP

- ATLAS Storegate: transient data store (collision event data)
- ATLAS Central Data repository
- StoreGate::Record(DataContainer, Key)
- StoreGate::Retrieve(Key, DataTypeOfContainer)
- StoreGate manages all EventData
- Optional: stored data can be made const.