Building Efficient Data Planner for Peta-scale Science

Michal Zerola[1] Jérôme Lauret[3] Roman Barták[2] Michal Šumbera[1]

[1]Nuclear Physics Institute, Academy of Sciences, Czech Republic

[2]Faculty of Mathematics and Physics, Charles University, Czech Republic

[3]Brookhaven National Laboratory, USA

ACAT 2010, Jaipur

Outline

1. *Introduction*
    - Motivation
    - FAQ
    - Optimization
    - Requested features
2. *Implementation*
    - Architecture
    - Planner
        - Requirements
    - Database schema
    - Data Mover
3. *Show case*
4. *Conclusions*

Why planning and scheduling in distributed environment?

Exploiting remote sites and centers implicitly opens the question:

How to handle, control and efficiently use the resources?

- balance between being fair to the users and optimizing utilization
- random and uncoordinated access to the resources will hardly be optimal

Why planning and scheduling in distributed environment?

Exploiting remote sites and centers implicitly opens the question:

How to handle, control and efficiently use the resources?

- balance between being fair to the users and optimizing utilization
- random and uncoordinated access to the resources will hardly be optimal

Our research tackles the efficient data movement:

- decoupling the data movement from job processing first

Why planning and scheduling in distributed environment?

Exploiting remote sites and centers implicitly opens the question:

How to handle, control and efficiently use the resources?

- balance between being fair to the users and optimizing utilization
- random and uncoordinated access to the resources will hardly be optimal

Our research tackles the efficient data movement:

- decoupling the data movement from job processing first

Current goal

Create mechanism for efficient and controlled way of moving datasets (replicated) to the destinations in the fastest way

Why planning and scheduling in distributed environment?

Exploiting remote sites and centers implicitly opens the question:

How to handle, control and efficiently use the resources?

- balance between being fair to the users and optimizing utilization
- random and uncoordinated access to the resources will hardly be optimal

Our research tackles the efficient data movement:

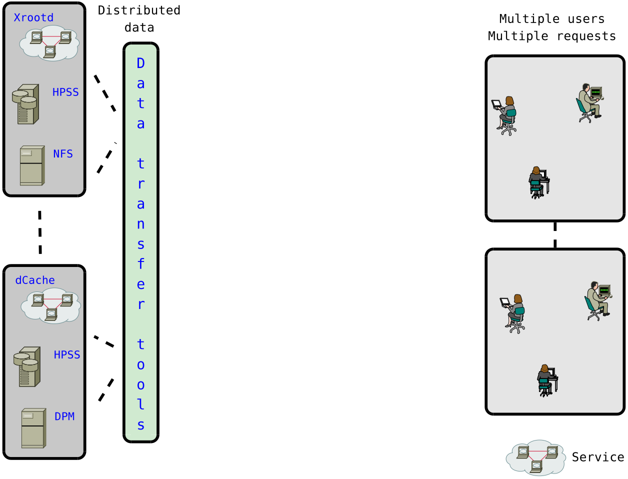
- decoupling the data movement from job processing first

Current goal

Create mechanism for efficient and controlled way of moving datasets (replicated) to the destinations in the fastest way

⇒ combination of deliberative and reactive planning ⇐

Situation



Distributed data

Xrootd

HPSS

NFS

dCache

HPSS

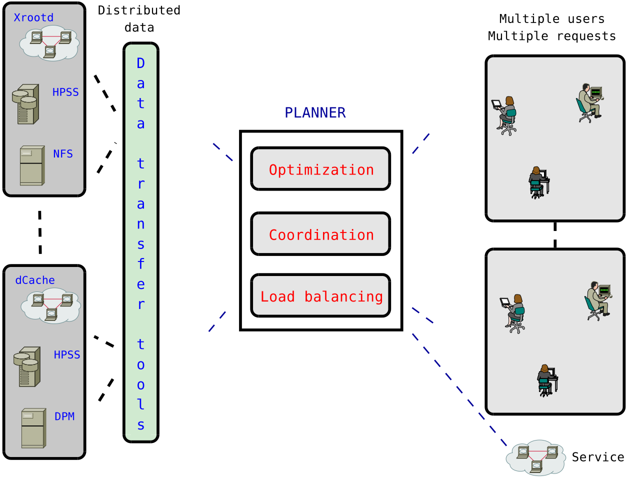
DPM

Data transfer tools

Multiple users
Multiple requests

Service

Goal


Xrootd
HPSS
NFS
dCache
HPSS
DPM
Distributed
data
Data transfer tools
PLANNER
Optimization
Coordination
Load balancing
Multiple users
Multiple requests
Service

FAQ

Are you building another N^{th} data transfer tool?

- No, we are building a mechanism sitting between users and existing efficient data transfer tools providing control, optimization and load-balancing.

Are you mirroring the topology and characteristic of the full network in your model?

- No, the model is based on approximation of the latencies/bandwidth and is from its startup point self adaptive to the environment.

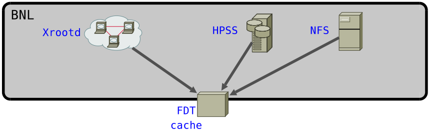
Can I use my own planner or fair-share policy?

- Yes, all decision-making modules are separate independent plugins.

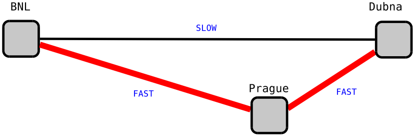
Optimization

Two levels of optimization:

1. Among **data services** (sharing the data)



2. Among sites - **data centers** (geographically spread)



Requested features

Control:

- respect different user priorities and usage history
- support any queue-based fair share policy
- provide estimates and status for the users

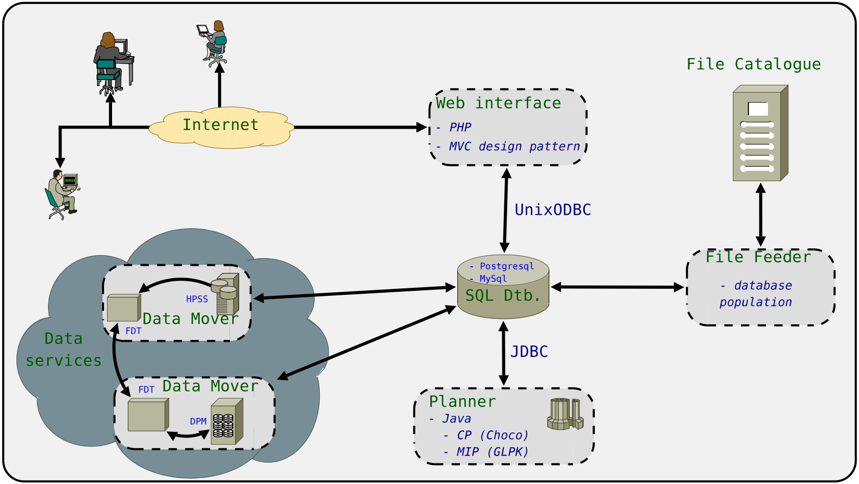
Load balancing:

- prevent uncontrolled access and overloading of resources
- load balancing of storage elements and network

Adaptability:

- proper balance between *reactive* and *deliberative* planning
- adapt to the network or service changes automatically

Architecture

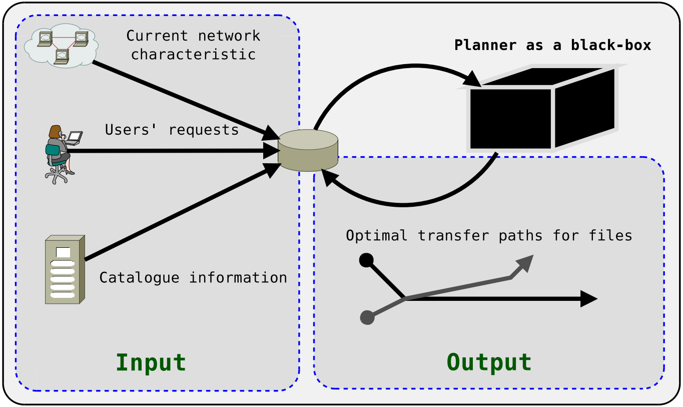


Planner

Constrained based, two approaches:

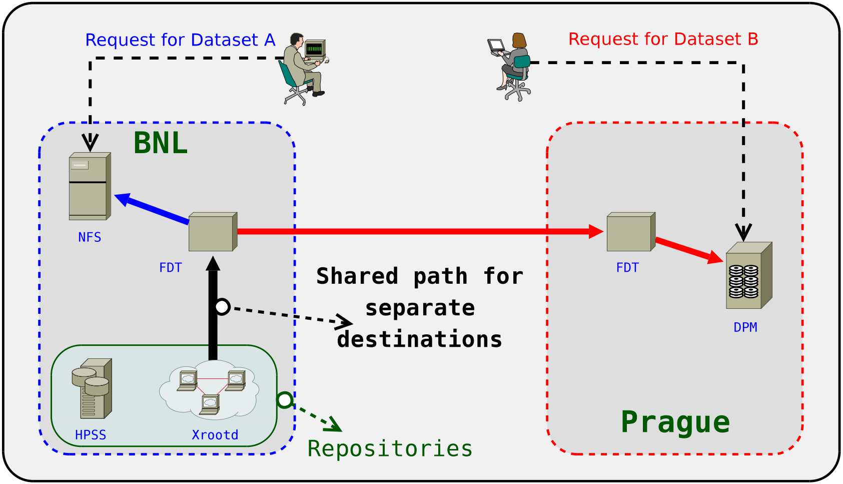
- Constraint Programming (Choco)
- Mixed Integer Programming (GLPK)

Fast, short-term deliberative planning



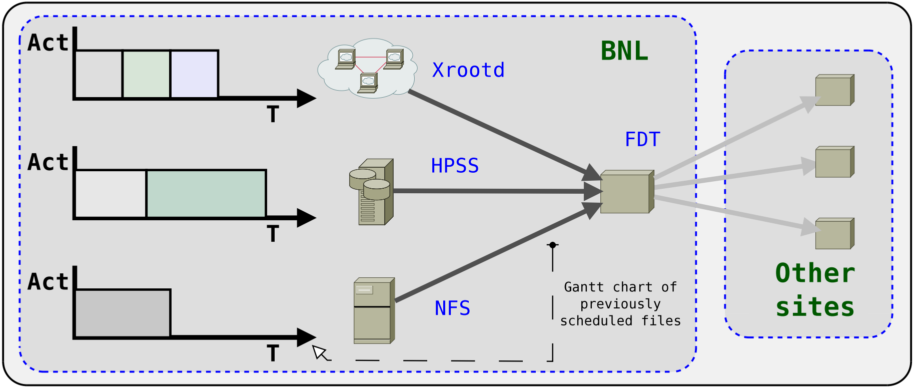
Requested features

- resources should be used effectively
- objective: **minimize time** to bring files to the users



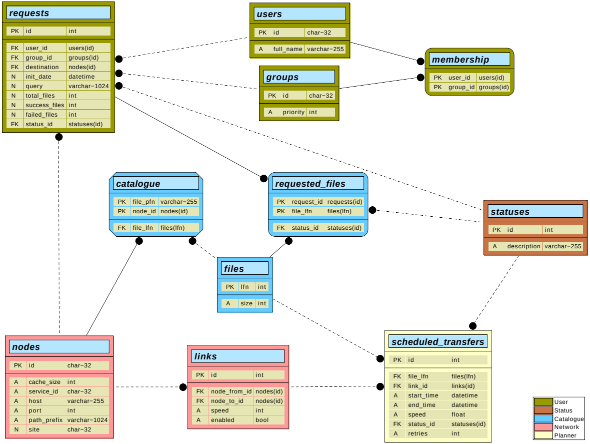
Requested features (cont.)

- use information about links usage from previously scheduled transfers
- avoid creating bottlenecks



Database schema

- most exposed parts: $10^5 - 10^6$ records in STAR
- support for *MySQL (InnoDB)* and *Postgresql*



requests

PK	id	int
FK	user_id	users(id)
FK	group_id	groups(id)
FK	destination	nodes(id)
N	init_date	datetime
N	query	varchar−1024
N	total_files	int
N	success_files	int
N	failed_files	int
FK	status_id	statuses(id)

users

PK	id	char−32
A	full_name	varchar−255

groups

PK	id	char−32
A	priority	int

membership

PK	user_id	users(id)
PK	group_id	groups(id)

catalogue

PK	file_pfn	varchar−255
PK	node_id	nodes(id)
FK	file_lfn	files(lfn)

requested_files

PK	request_id	requests(id)
PK	file_lfn	files(lfn)
FK	status_id	statuses(id)

statuses

PK	id	int
A	description	varchar−255

files

PK	lfn	int
A	size	int

nodes

PK	id	char−32
A	cache_size	int
A	service_id	char−32
A	host	varchar−255
A	port	int
A	path_prefix	varchar−1024
N	site	char−32

links

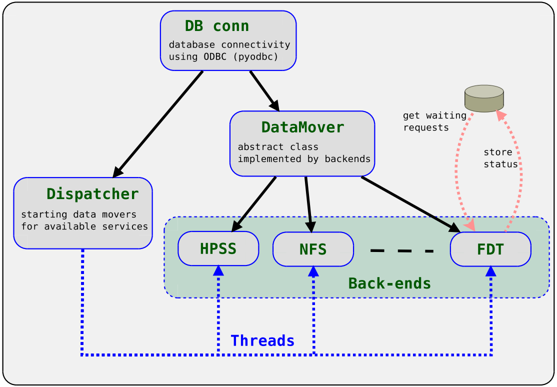
PK	id	int
FK	node_from_id	nodes(id)
FK	node_to_id	nodes(id)
A	speed	int
A	enabled	bool

scheduled_transfers

PK	id	int
FK	file_lfn	files(lfn)
FK	link_id	links(id)
A	start_time	datetime
A	end_time	datetime
A	speed	float
FK	status_id	statuses(id)
A	retries	int

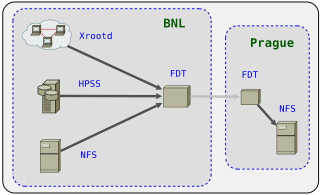
Data Mover

- **distributed** component, using efficient and existing transfer tools
- running at each computing center, implemented in Python
- back-ends for available services realize the transfers
- works in a reactive way, following the computed plan



Show case - setup

- moving data set to the single destination - **NFS** location in Prague
- every file from the dataset available at **HPSS**, **NFS** and **Xrootd** service in BNL



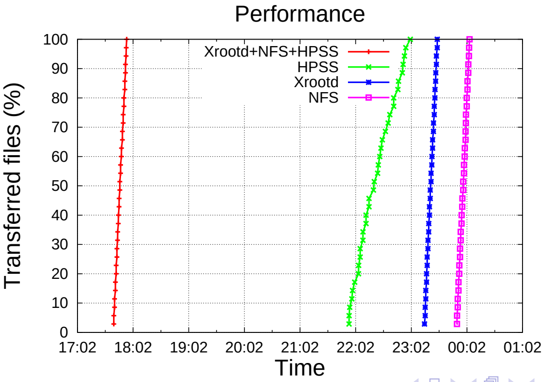
- matrix for success is to be limited by *WAN* speed alone

Show case - comparison

Planner was set up to reason about:

1. all services (HPSS, NFS, Xrootd) as possible data sources,
2. only HPSS (slow),
3. only NFS (fast),
4. only Xrootd (fast)

The use of resources was: HPSS - 19%, NFS - 38%, Xrootd - 43%



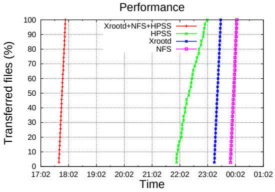
Show case - conclusion

Remarks:

- Files are usually not all on NFS (central storage reserved for ongoing data production in STAR, not past data production series)
- Users would have to grab files from Xrootd or HPSS

  Xrootd would create a load and impact analysis - our system provides **immediate** relief without sacrifice of the transfer plan time

  HPSS stress and frequent access to **HPSS** hence tape access could be damaging to tape (wear-out) + competes with other access (production sync to HPSS)
- Our system balances resources in an adaptive fashion.



Ultimately

Utilizing all resources brings the same performance as relying only on the fastest one (NFS/Xrootd) while bringing load-balancing, control and redundancy.

Conclusions

Status:

- planner, database, web interface are prepared and functional
- performance of the pure planner extensively studied and tested in simulated environment
- all components are functional and installed in STAR, currently running tests

Perspectives:

- implement multi-site transfers: we expect similar benefits in balancing
    - immediate relief to the *Tier-0* center
    - self-adaptive capabilities will determine the best transfer path
    - data integrity benefits for "free"

Summary:

- the concept of controlled and efficient data movement brings:
    - better efficiency due to **intelligent planner**
    - controlled **coordination**
    - **load-balancing**

Conclusions

Status:

- planner, database, web interface are prepared and functional
- performance of the pure planner extensively studied and tested in simulated environment
- all components are functional and installed in STAR, currently running tests

Perspectives:

- implement multi-site transfers: we expect similar benefits in balancing
  - immediate relief to the *Tier-0* center
  - self-adaptive capabilities will determine the best transfer path
  - data integrity benefits for "free"

Summary:

- the concept of controlled and efficient data movement brings:
  - better efficiency due to **intelligent planner**
  - controlled **coordination**
  - **load-balancing**

Thank you!