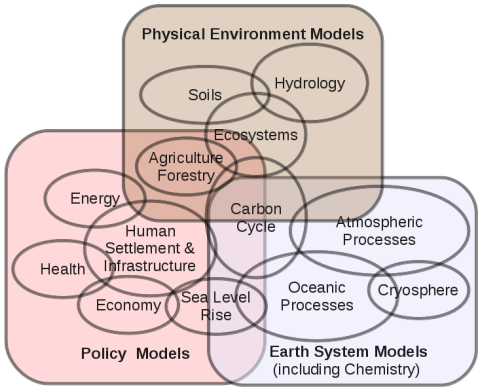
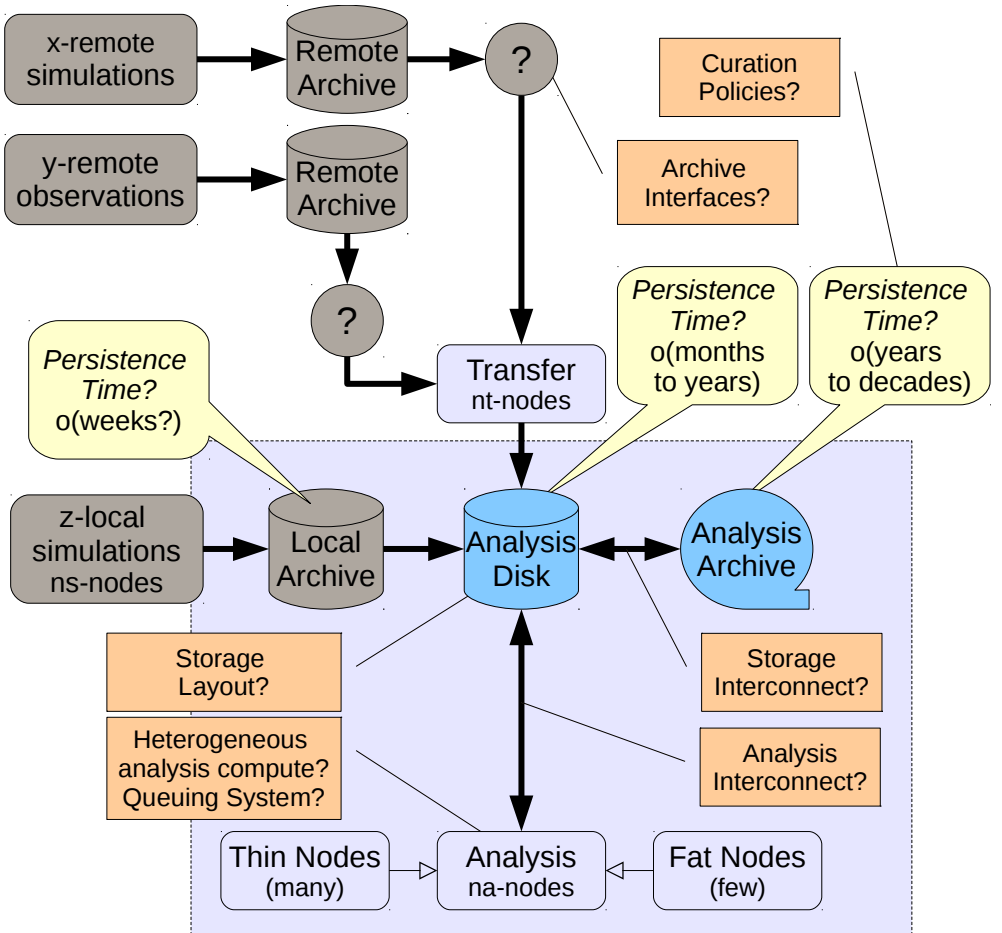


Many interacting communities, each with their own software, compute environments etc.

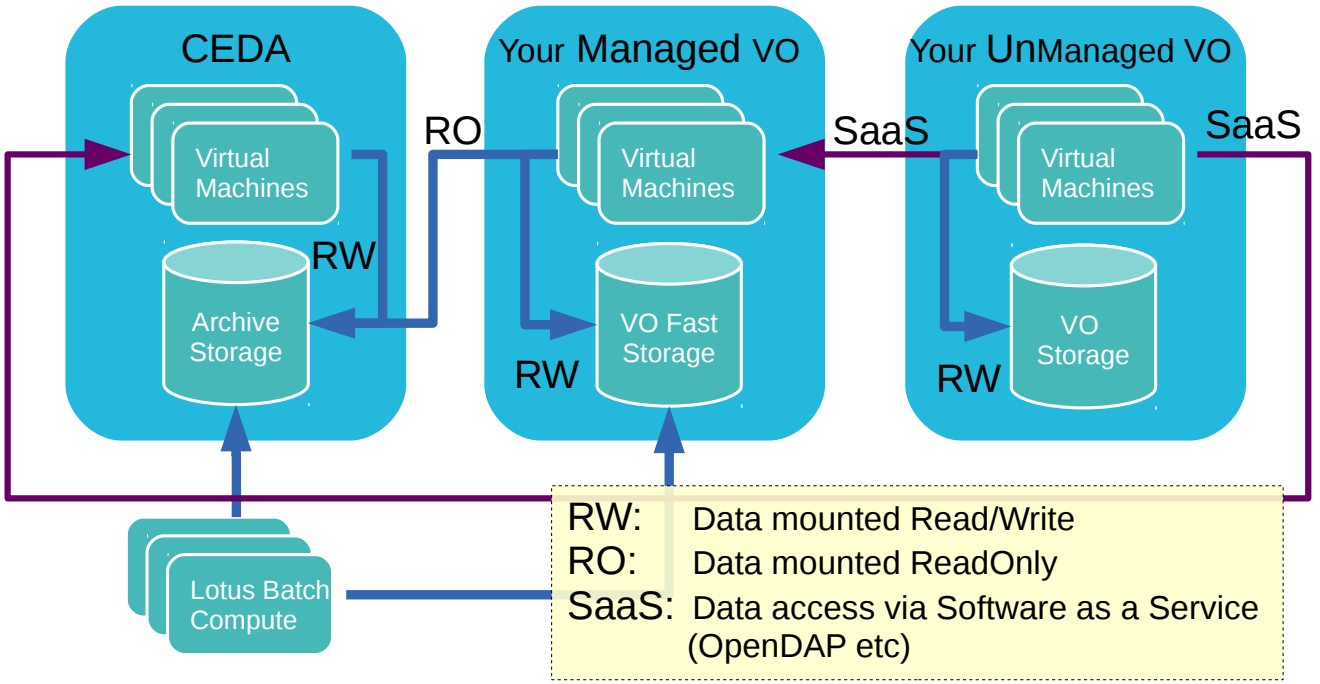
16 PB Fast Storage
1 PB Bulk Storage, Elastic Tape
4000 cores
(hypervisors & batch cluster)

JASMIN SuperData Environment



Why is this BDEC? Because in the petascale era, we're handling petabytes of storage with terabytes in each of hundreds of workflow. In the exascale era, we'll have exabytes of storage, with petabytes in hundreds of workflows.

Objective is to provide an environment with high performance access to curated data archive **and** a high performance data analysis environment!



Curated environment one virtual organisation within o(100) such virtual organisations. Key issues include:
(1) how to provide high performance data access and analysis in the managed environment for multiple users, multiple workflows, intersecting in some of the data,
(2) between unmanaged (infrastructure as a service) and the data held in (our) managed environment, and
(3) data growth that exceeds the Kryder rate (volume/bandwidth etc).

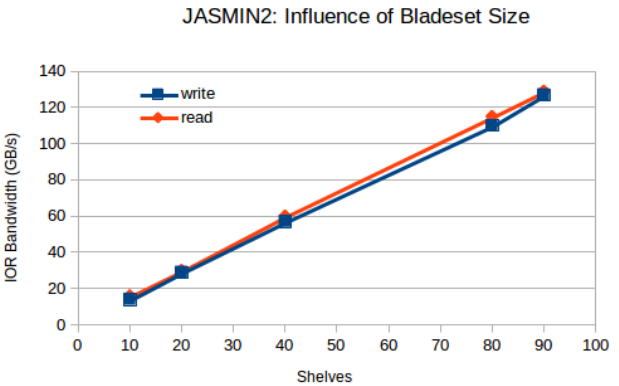
The seven deadly sins of ~~cloud~~ computing research
Schwarzkopf, Murray, Hand
Hotcloud, 2012

Pick five, all in play:

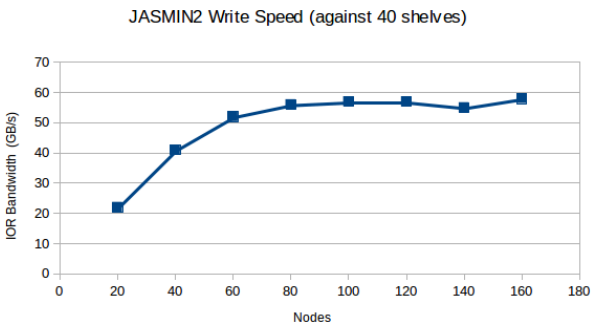
- *Unnecessary distributed parallelism:* We need to support (nicely) high memory and other nodes inside our environment.
- *Assuming performance homogeneity.* This is a real problem for us in a mixed VM/batch environment ... Help.
- *Forcing the abstraction (Map-Reduce, HADOOP or bust)* We avoid this by having a parallel file system, but how do we know we are getting value?.
- *Unrepresentative workloads.* We really don't know how to optimise our jobs (yes, we can give people exclusive access to nodes, but it's harder to give them exclusive I/O bandwidth).
- *Assuming perfect elasticity.* We haven't worked out how to schedule to use our resources, or how to cloud burst properly.

We need work on understanding all these things

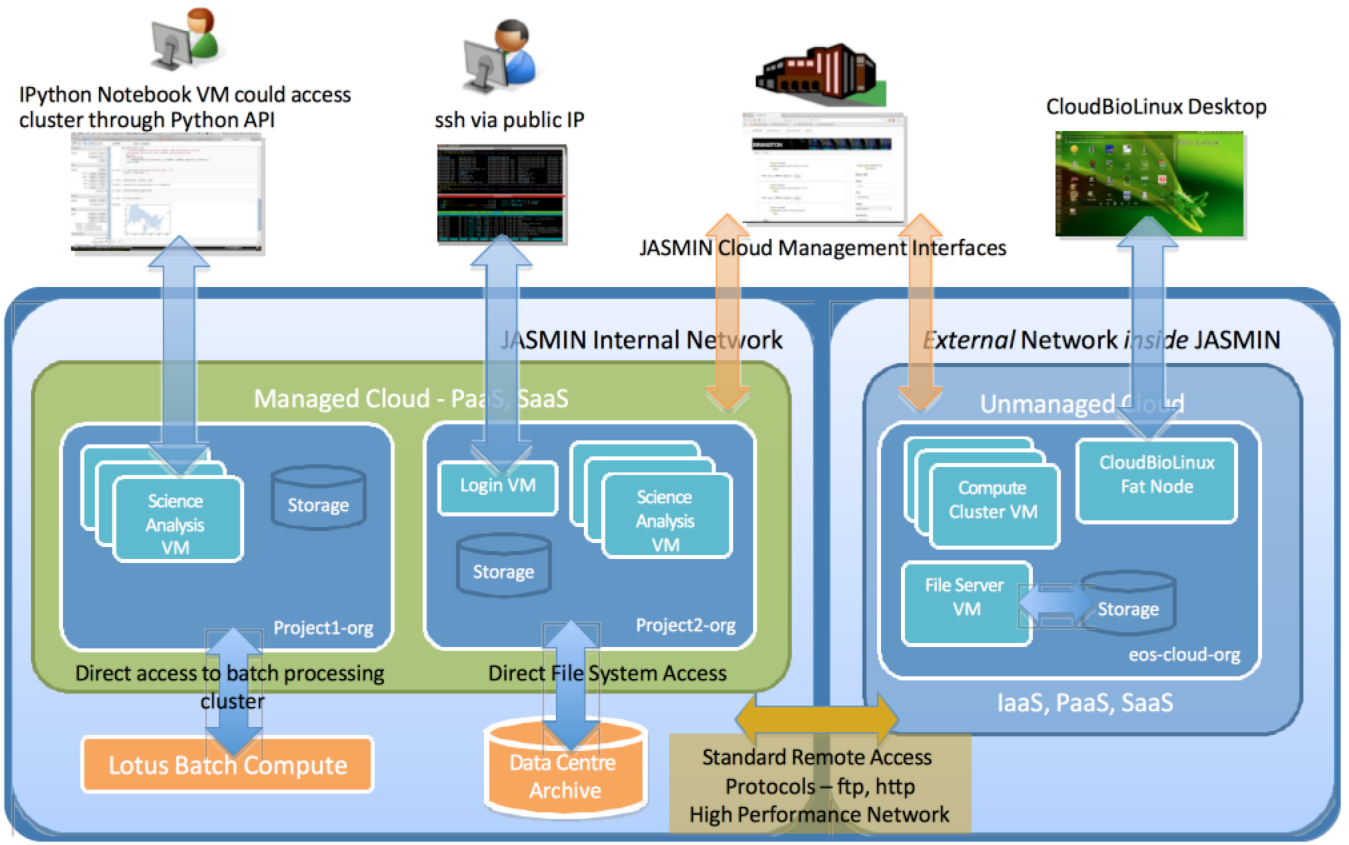
Pick one issue: I/O ~~optimisation~~/control



Do we understand the performance at the user/app level?
We can break our file system up into pools ("blade sets") in Panasas.
Give communities access to resources on one blade set.
Now their I/O does not interfere with VOs using other blade sets.



Issues:
— This isn't very flexible! We can still nail a PB bladeset with 80 nodes! How do we get more and flexible I/O parallelisation?
— When we run out of physical space for disk, how are we going to efficiently use tape in our workflow?



This currently works because we have spare capacity, and few users in the un-managed cloud and the ipython notebook environment.

We don't know how to do the scheduling here, the hypervisior/VM paradigm is banging up against the batch system job. Interactive is banging up against resource. The sixth deadly sin: there is not perfect elasticity. We are offering cloud bursting (to Amazon and Azure, we hope), but then there needs to be more work on data pipelines.