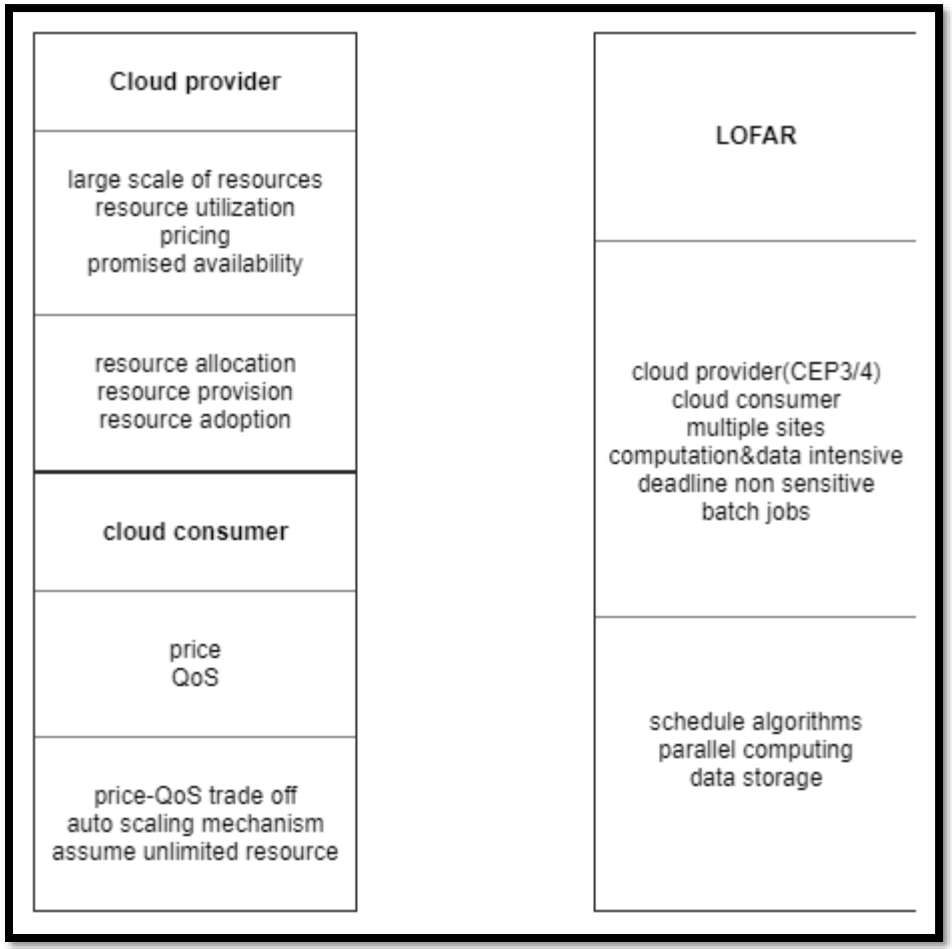
Self adjusted auto provision system at resource level

Weekly report 15th April

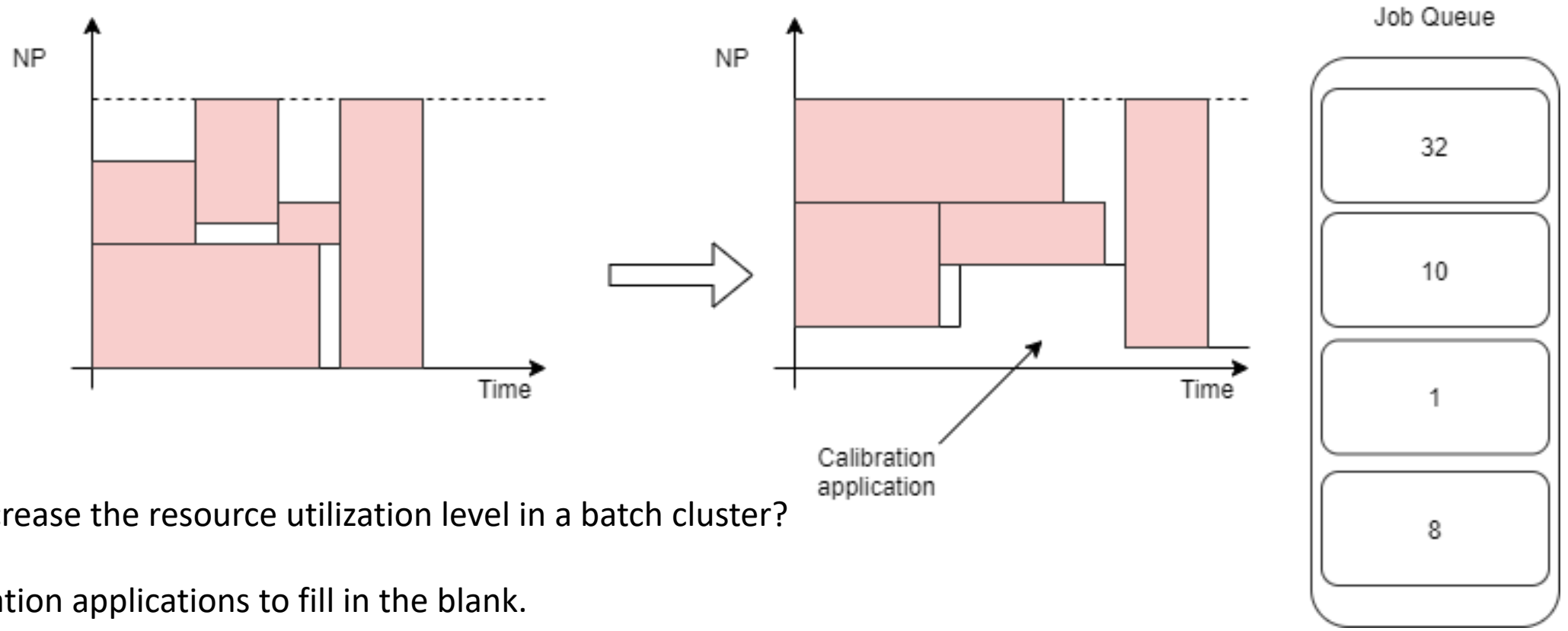
You Hu

Recap – Background & Requirements



It is close to the case of private cloud

Recap – issue and research question

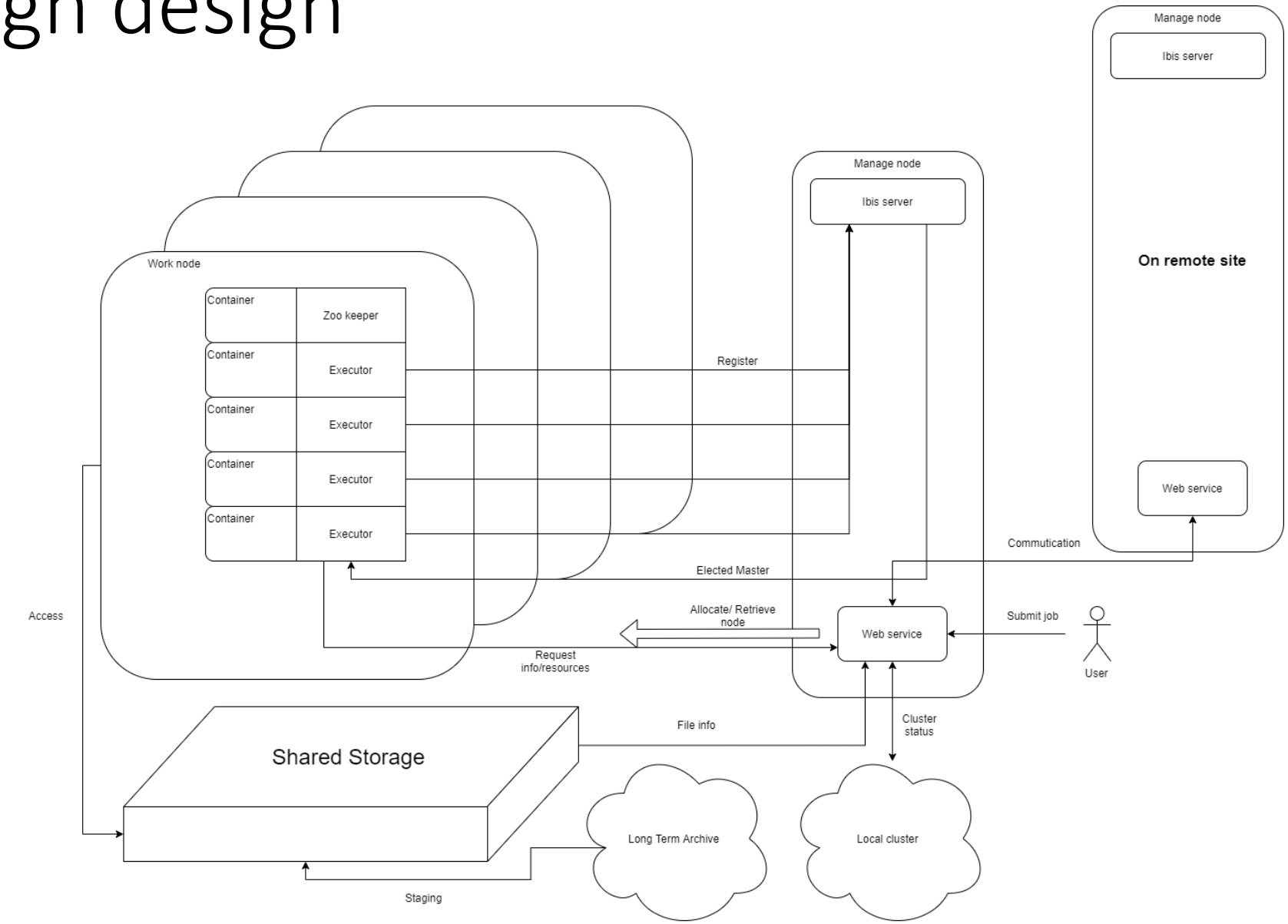


How to increase the resource utilization level in a batch cluster?

Use calibration applications to fill in the blank.

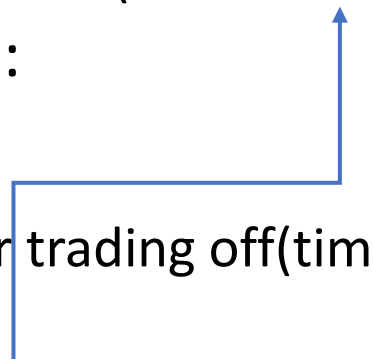
It requires: auto scaling/provisioning;

Recap – Rough design

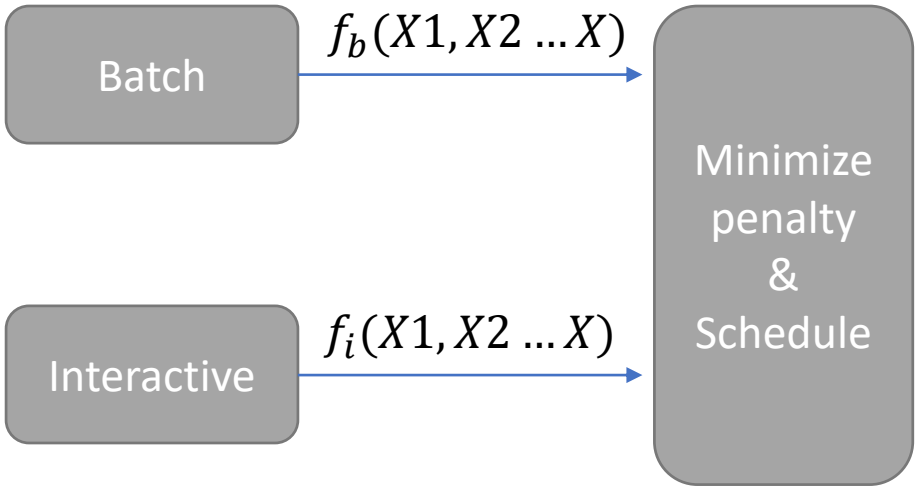


Questions statement

- Distributed algorithm/architecture: Master – worker
  - Communication: IPL -> Ibis server(s-for backup)
- Resource management algo:
  - Estimate job execution time
  - Schedule->SLURM+Xenon
  - Dynamic adjust: a metrics for trading off(time to start and shut down nodes)
- Fault tolerance
  - Node manage: master->re-election; worker->task persistence
  - Tasks(NO snapshot): Zookeeper/Redis/any lightweight distributed database
- Data locality: move comp rather than move data
  - Remote sites
  - Shared storage
  - Data aware(?)



Existed research



The penalty calculation is based on SLA

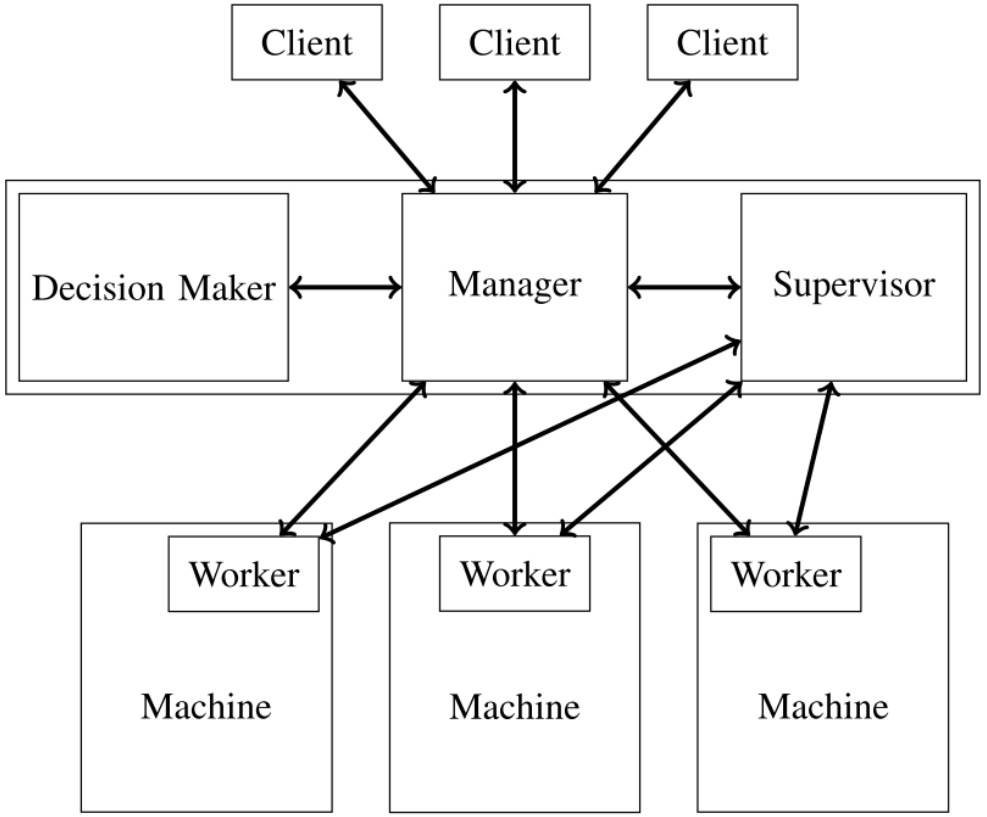
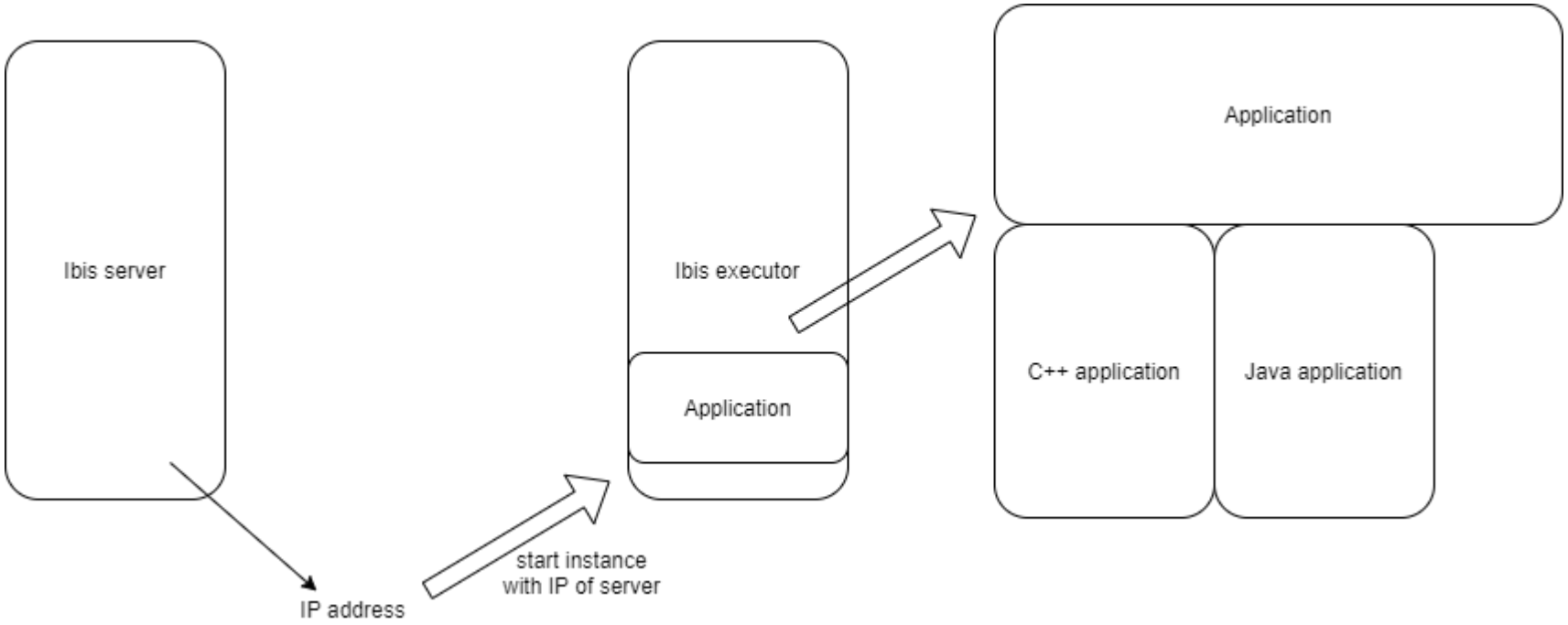


Fig. 1: System Architecture

Chang, T. W., Lin, C. C., Liu, P., Wu, J. J., Shih, C. C., & Huang, C. W. (2016). Resource provision for batch and interactive workloads in data centers.

Current experiment – Ibis + JNI + container

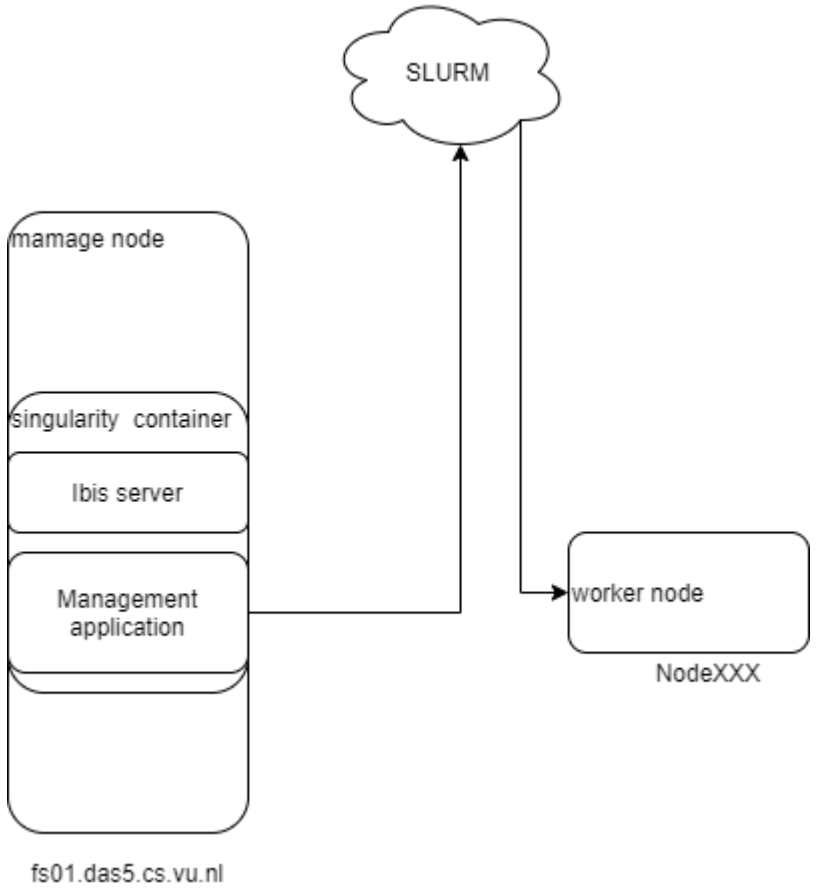


The JAVA and C++ applications have been compiled when the images of executor built(by Gradle)

Current experiment – Xenon + SLURM

Problems:

- No container now: port mapping(where is ibis server?)
- Can not connect head node: × location = "ssh://localhost“



Other topics

- Remarks on the design
- Ibis and Xenon, how to excavate the potential of them on this application
- Schedule management of SLURM
- Next phases
  - Literature study
  - Experiments