Motr
Layout Access Plan

Nikita Danilov, Andriy Tkachuk, Mar 2021

Why?

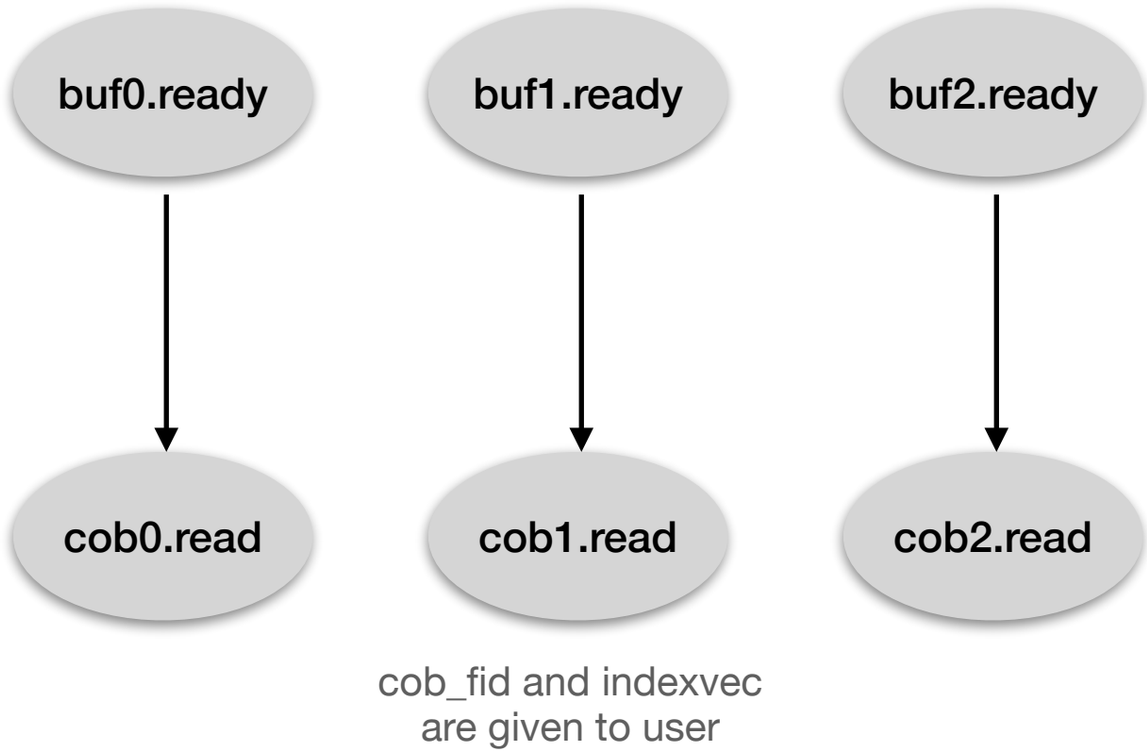
- For ISC (In-Storage Compute, aka Function Shipping) who needs to know where the user data is located
- We need generic and fault-tolerant API
- I/O code can be rewritten based on it (later) and greatly simplified

Plan is a graph of instructions

- Plan is just a directed graph of instructions (plan operations - **plops**) the user (ISC or I/O code) has to run to access the data
- There are several types of plops, like
    - read or write the cob data (unit) from/to server or
    - call a function (like parity calc or compress/decompress)
    - the data is ready - take it
- User gets plops from the plan and executes them according to the dependency order calling plop_done() for each plop

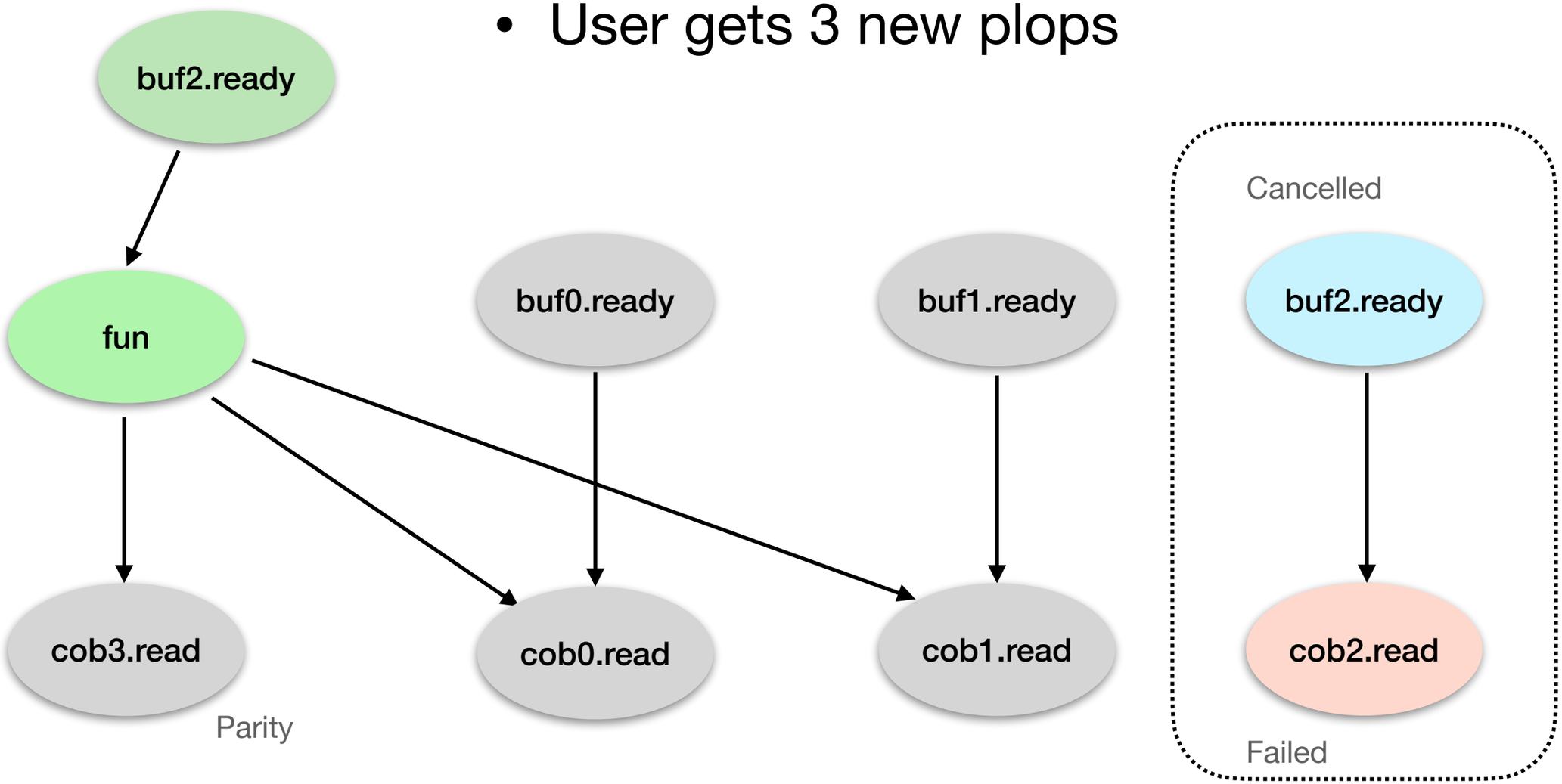
Simple read

- User gets 3 plops to read data units and
- 3 plops with the data ready
  - Note: the data will be actually ready only after user reads the data from server and calls plop_done() for the read plop



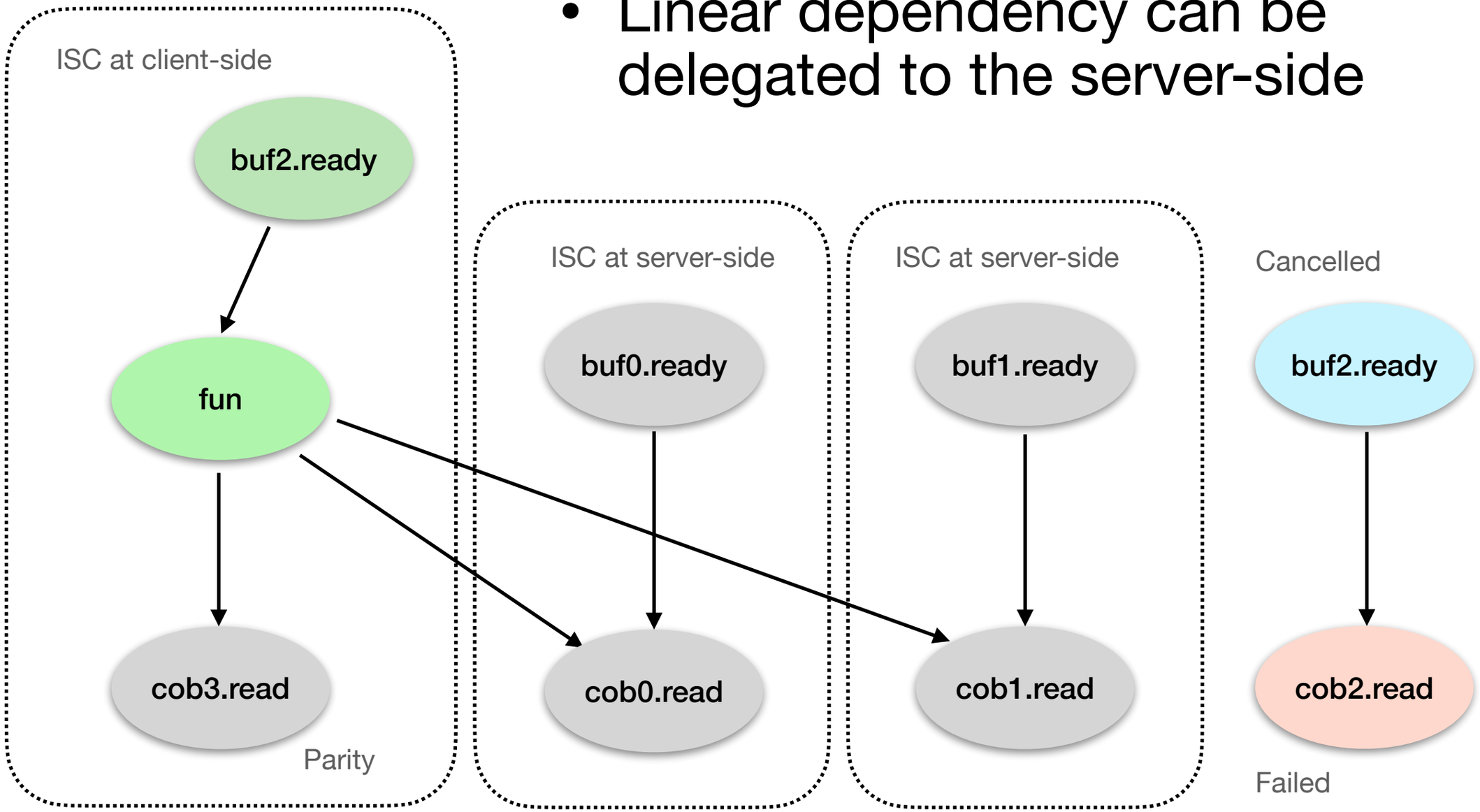
Failed read

- cob2.read is failed, user calls plop_done() with error in pl_rc
- Plan changes dynamically to degraded read, buf2.ready is cancelled
- User gets 3 new plops



ISC usage

- ISC-code must analyse the dependencies between plops:
- Linear dependency can be delegated to the server-side



Questions?

- PR - https://github.com/Seagate/cortx-motr/pull/411