3. Most files are modified by accident.

2. File sizes huge by traditional standards.

- Apple bugs, o/s bugs, human error

- Shrink/expand etc.

- Client command components (client gui)

1. Command for Ice, some

- Revisions for new FS

- Design of FS

- CSC535 Fall 2015
But, on the other hand, if we consider the consistency of the system, the number of API calls is also crucial.

Approaches to consistency:
- AV1 for common case of multiple users
- Relax consistency requirement

e.g. Co-designing API and app behavior.
Assumptions:
1. System is built from inexpensive commodity components
2. Constant monitoring for faults
3. Recover quickly from these faults on routine basis

Principle:
Optimize for common case.
1. Small files must be supported but need not be optimized for.
2. Multi-GB files should be managed effectively.
- small random units can be very
  - once written, seldom modified
  - 8+.20 or more
  - to file
  - many large-seg. write that appears

- Write:
  - 8 logical track.
  - effort (450.73) 0.65
  - part, conflict, early unused reads
  - random offset.

- Typically from KG area.
- Typically from KG area.
- Success reads from 1 to more.
- Individual reads in 1H or more.

Reads:

3. Workload
do in with at a high rate. 

A phone place a banquet in an Proceed. 

Low. 

High. 

The program. 

Overhead. 

A theme with minimum. 

Dry to appear. 

As many producers will continually. 

As many may marry. 

Fills as producer consumer gloss. 

Anaps. 

For multiple claims that are consistently approved. 

In effect. 

Inferred model. 

4.
By serendipity, our configuration can be optimized using decon.
for example, in plant focal-manner

I'm curious to uncover by now, else

This [paradoxically] heart-beat: adaptation

- Stale.
- Soft.
- Black

- Paste - back up - Quinn change season
- Notate

Check location information encirclenun