Architecture of Obliterate

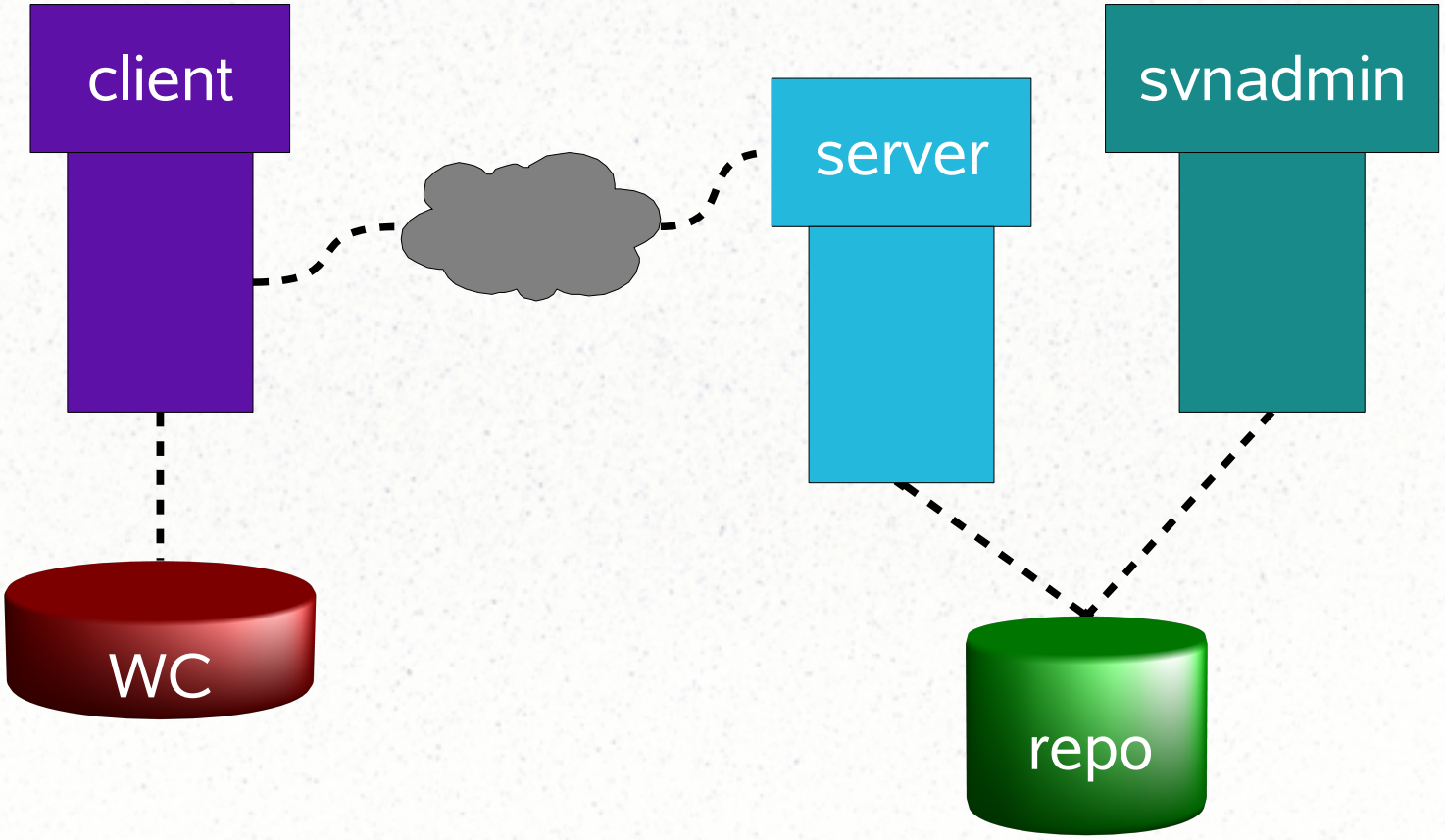
Julian Foad

WANdisco

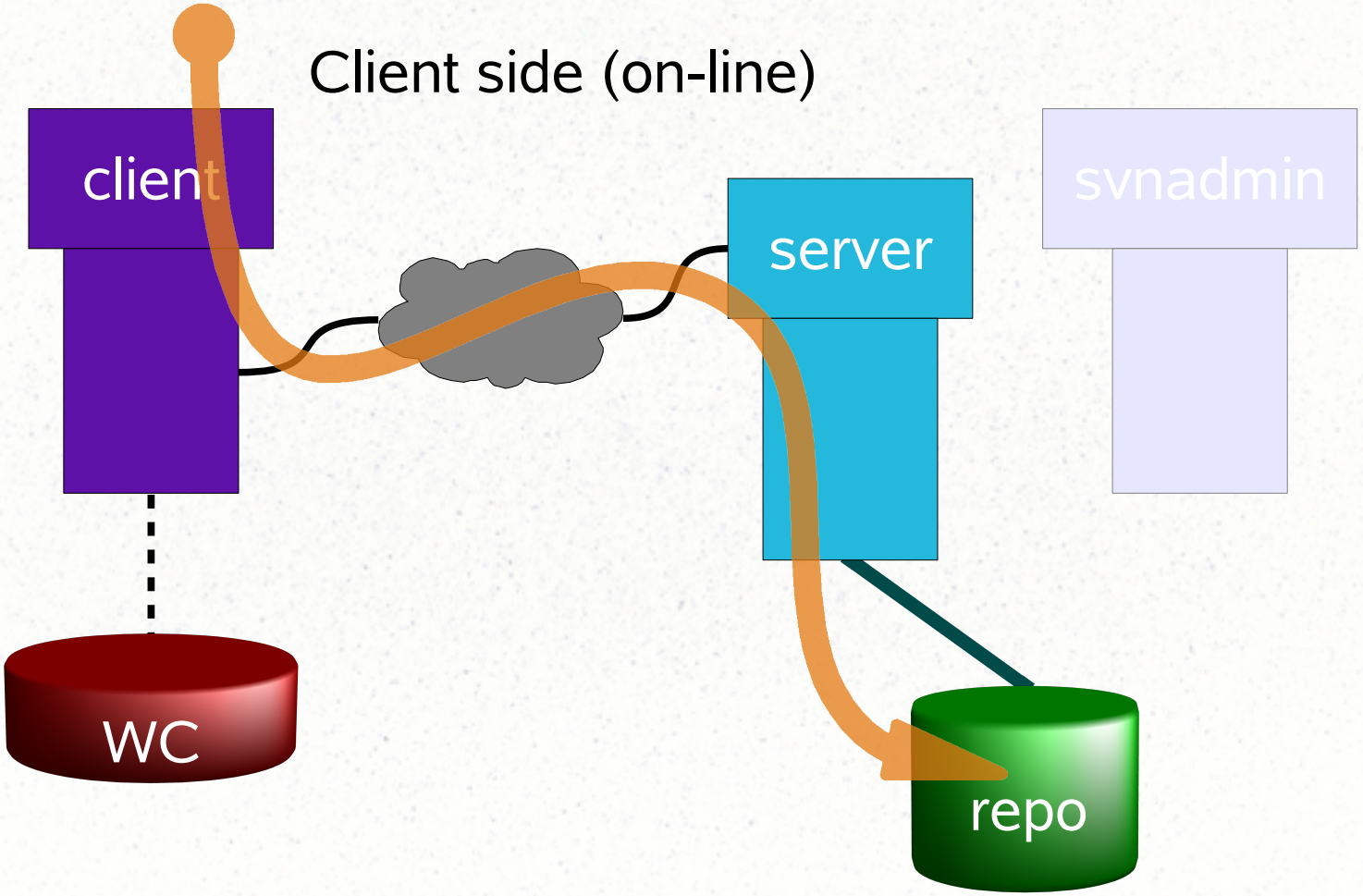
Overview

- Server or Client
- Tell Server to Obliterate r50?
- DB Transactions
- Authorization
- Client Issues
- Protocol Extensions

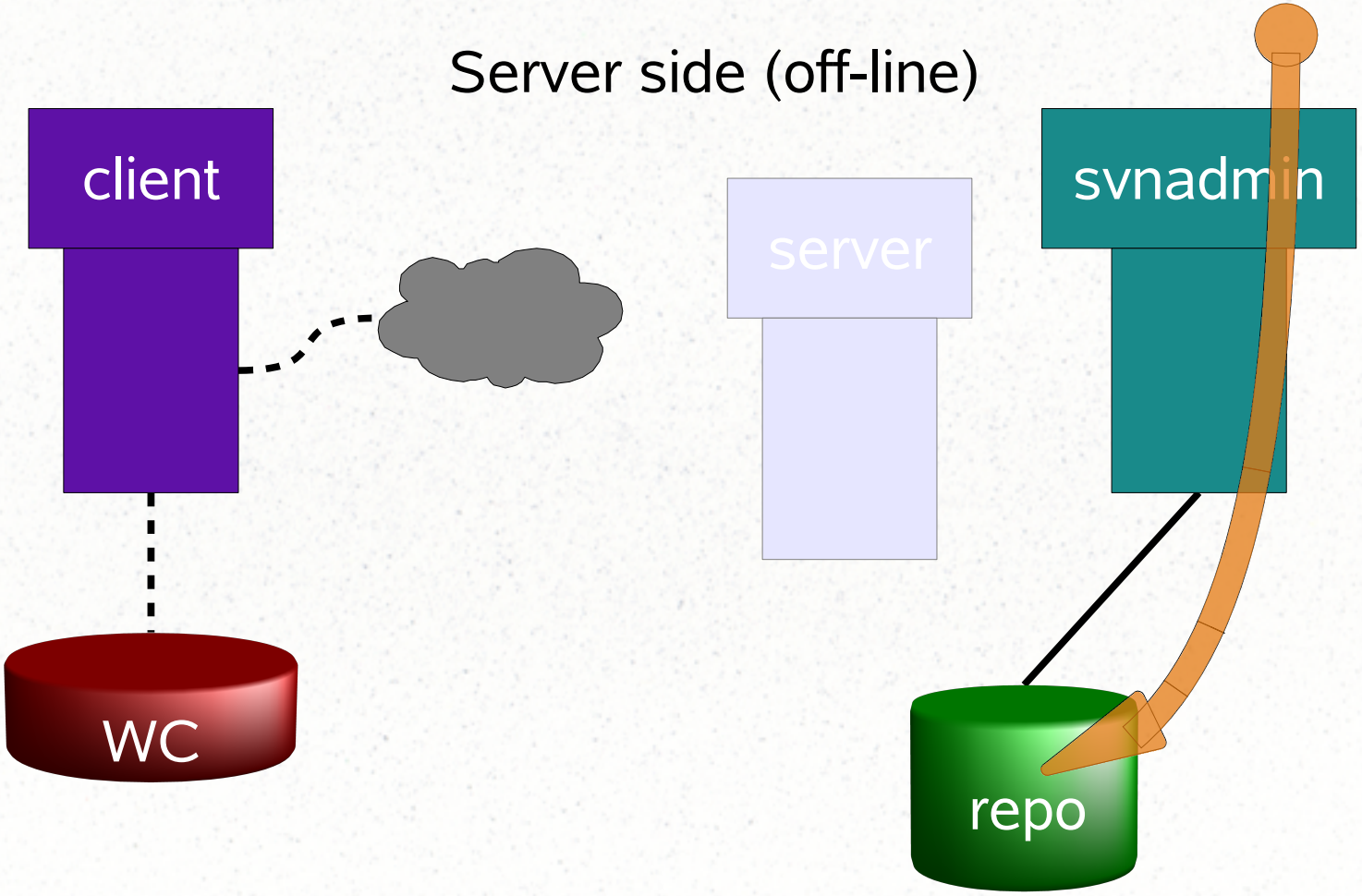
Server or Client (1/3)



Server or Client (2/3)



Server or Client (3/3)



Tell Server to Oblit r50? (1/2)

Present client-server protocol

- says “create a **new** HEAD revision”
- cannot say “**modify** revision 50”
- server doesn't know how to.

Tell Server to Oblit r50? (2/2)

We need

- new function in **server** libraries

  perform a “primitive” obliteration

- new command in **network protocol**

  describe a “primitive” obliteration

- new UI in **client**

  interpret what the user wants

  send obliteration commands

Normal Transaction (1/1)

Illustrate the basic new-head transaction

- Repo has r0, r1, r2=HEAD

  Client knows what repo has (up to r2)

- Client tells server to

  CONSTRUCT a tree based on r2

  COMMIT this tree as the new HEAD

  - server checks HEAD is r2 or compatible
  - server links the txn in as r3

Obliterate Txn (1/5)

Shape:

“replace old **revision N** with ...”

- Pro:

  same shape as existing transactions

- Con:

  some obliterations involve many revs

Obliterate Txn (2/5)

- Steps

  construct a replacement for r50

  guarantee **consistency**

  link in between r49 and r51

  destroy the old unlinked txn

Obliterate Txn (3/5)

Consistency

- check each node reference is valid
- check them all again in finalization

Obliterate Txn (4/5)

Finalize

- **check** all node references are valid
- **replace** the old r50 with this txn
- **destroy** the old unlinked txn

    mark orphaned nodes as invalid

    enables space recovery

    - doesn't necessarily recover space

Obliterate Txn (5/5)

Alternative shape:

“replace **history of object X** with ...”

- Pro:

  obliterate a node's history in one go

- Con:

  doesn't seem to fit the FS schema

Authorization

-
-

Client Issues

Coping with history changes

-

Protocol Extensions

-
-

Architecture of Obliterate

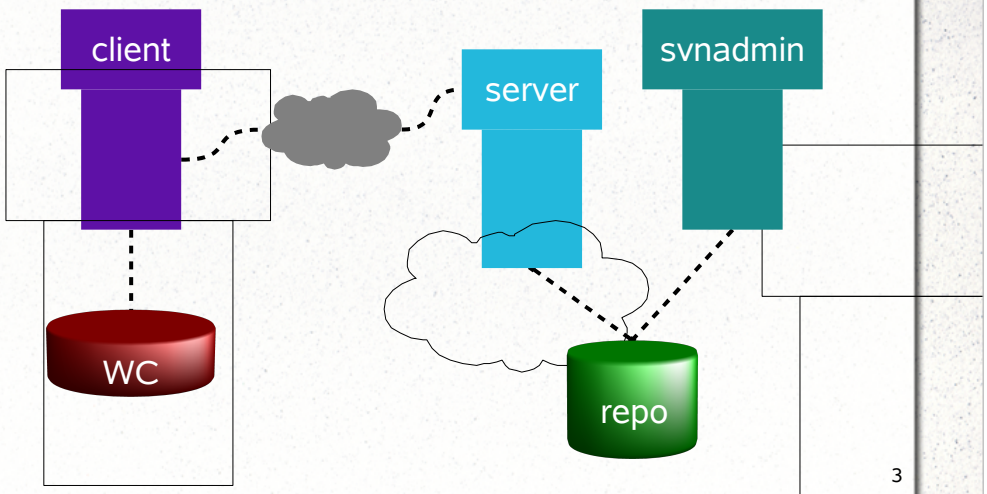
Julian Foad

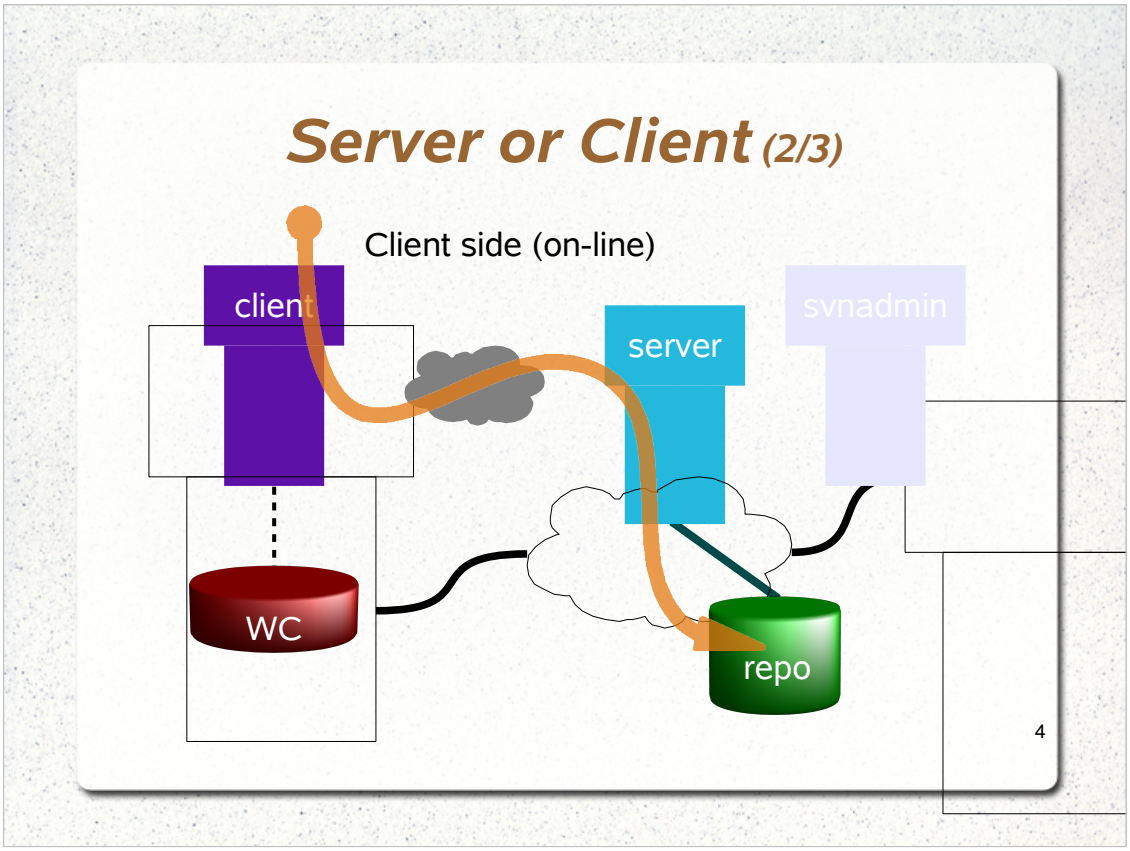
WANdisco

Overview

- Server or Client
- Tell Server to Obliterate r50?
- DB Transactions
- Authorization
- Client Issues
- Protocol Extensions

Server or Client (1/3)



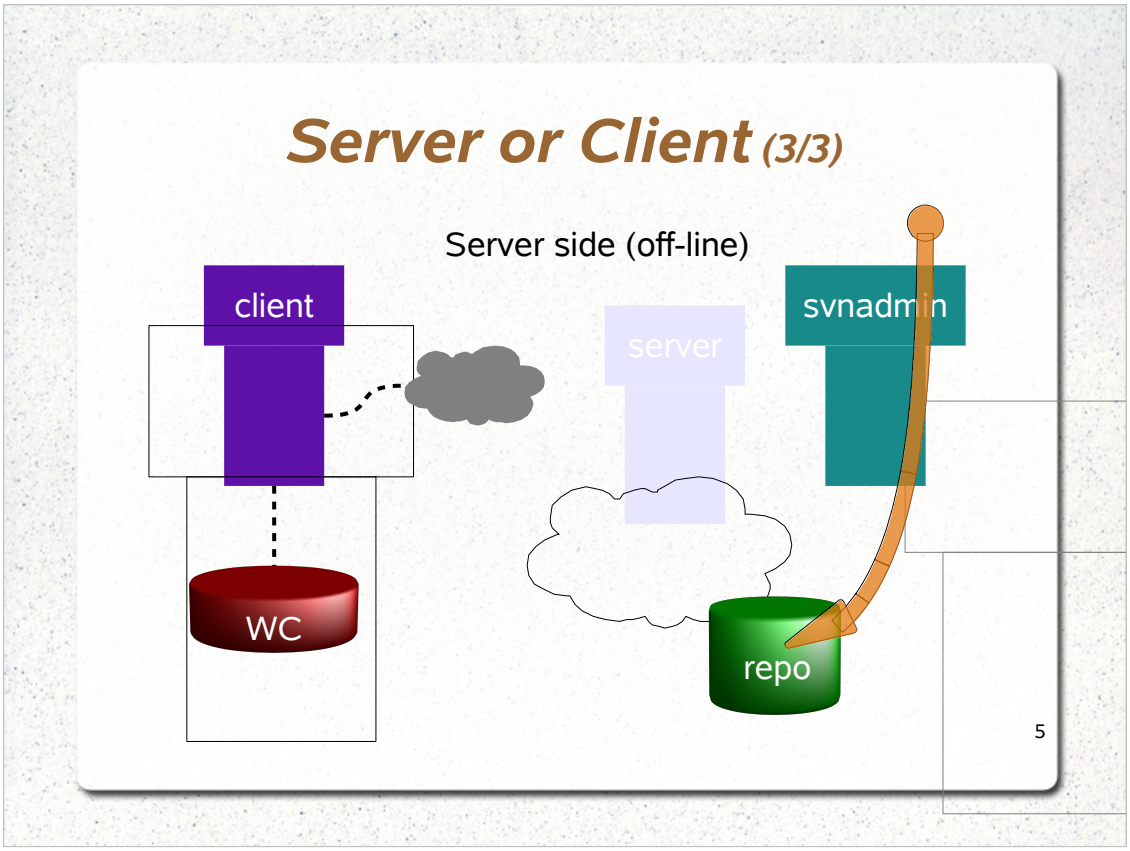


Repo is on line.

- MUST access repo through the running server
- full potential for all use cases
- extend the existing net protocols
  - could instead provide the server with a totally different interface, but doesn't seem sensible
- extend the existing client(s)
  - could instead write a separate client, but doesn't seem sensible

Paranoia:

- will likely want a server-side “obliterate on/off switch”, for when net authn/authz considered insufficient



Repo is off line

- This solution is suitable for planned maintenance only.
- Simpler design and implementation – no concurrency issues.

Tell Server to Oblit r50? (1/2)

Present client-server protocol

- says “create a **new** HEAD revision”
- cannot say “**modify** revision 50”
- server doesn't know how to.

Tell Server to Oblit r50? (2/2)

We need

- new function in **server** libraries

  perform a “primitive” obliteration

- new command in **network protocol**

  describe a “primitive” obliteration

- new UI in **client**

  interpret what the user wants

  send obliteration commands

Normal Transaction (1/1)

Illustrate the basic new-head transaction

- Repo has r0, r1, r2=HEAD

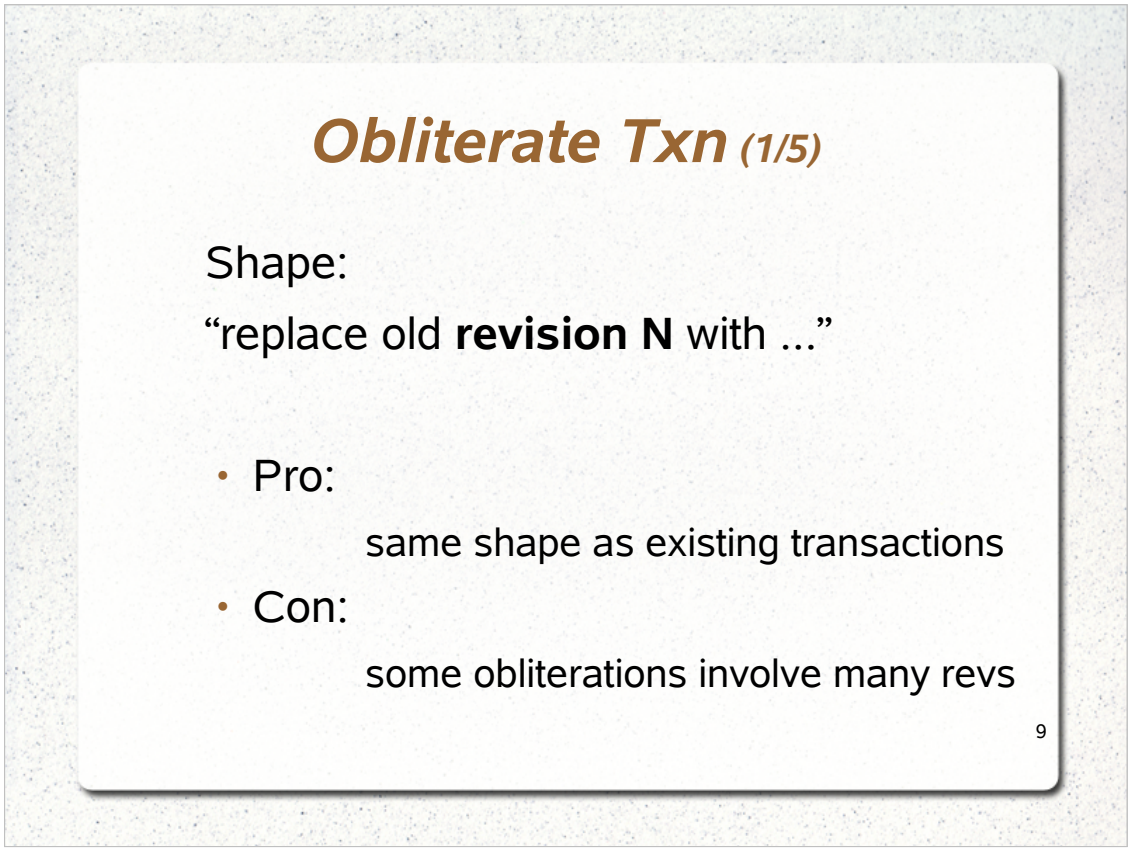
  Client knows what repo has (up to r2)

- Client tells server to

  CONSTRUCT a tree based on r2

  COMMIT this tree as the new HEAD

  - server checks HEAD is r2 or compatible
  - server links the txn in as r3



- Server needs to construct and commit a new kind of txn, one that changes an existing revision

Obliterate Txn (2/5)

- Steps

  construct a replacement for r50

  guarantee **consistency**

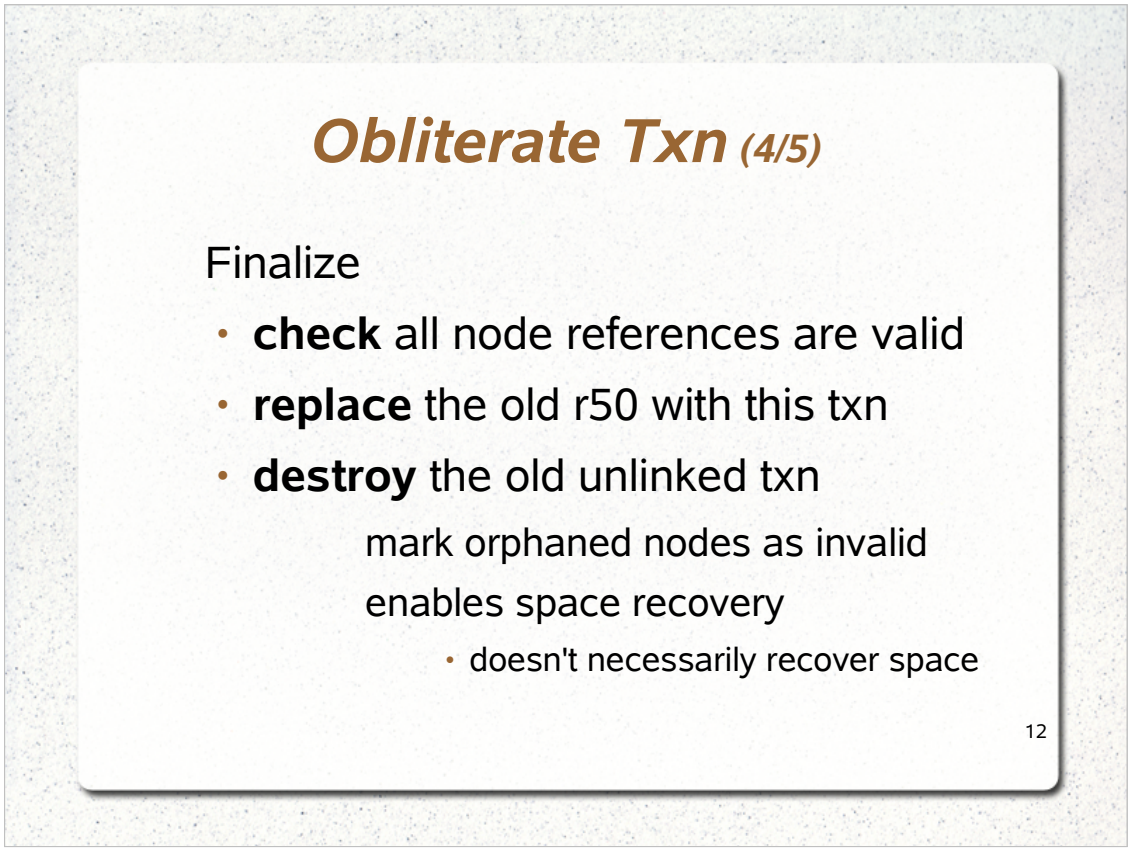
  link in between r49 and r51

  destroy the old unlinked txn

Obliterate Txn (3/5)

Consistency

- check each node reference is valid
- check them all again in finalization



The check needs to be done in normal transactions as well as in obliterate transactions.

To reduce cost of the check: an “obliteration serial number” (meta-revision of repository) could help. (Check again only if any obliteration happened since txn began.)

Obliterate Txn (5/5)

Alternative shape:

“replace **history of object X** with ...”

- Pro:

  obliterate a node's history in one go

- Con:

  doesn't seem to fit the FS schema

Authorization

-
-

Client Issues

Coping with history changes

Protocol Extensions

-
-