Architecture of 3storage

Page file server (PFS)

- Page file is a sequence of pages
- A *page* is a basic unit of processing
- All pages have the same length
- A *page identifier* (abbr. `pgid`) points to the beginning of the given page (offset of a page in page file)
- The page FS does not know anything about the contents of pages
- Architecture of Page file server
- Page server is an Erlang process
- Page file server accepts requests from many (ISAM) processes
- N-th page is accessed by reading|writting from|to the position n*page-size in db file
- Requests are placed in a queue and served one by one
  - Erlang process `pid` is stored for each request
  - (to-do) Does it make sense to have sessions (with a given process `pid`)?
- Buffer manager is serving as a cache
  - Page are read into buffer pool
  - Some page replacement strategy should be used
- Page server interface
- `read pgid N` command reads a sequence of `N` pages starting at the page `pgid`
- `write pgid N` command writes a sequence of `N` pages to the db file

ISAM index

- ISAM index is a key/value store
- Keys/Values are stored in data (leaf) pages
  - Keys are composed of instances of `K` subseteq {`S`,`P`,`O`}
  - A `value` can be either a pointer or a complete triple
- Index pages are storing keys/pointers to subtrees
- In the case triples are stored in data pages, we have a triple file organization
  - A triple file implements one of seven possible orderings of triples
- Architecture of ISAM index
- ISAM index is an Erlang module and process
- ISAM index is stored in a file of pages

  - ISAM index module includes a reference (`pid`) of a PFS Erlang process
- (to-do) To implement *cursors* in TP-AM we need to remember the position of *index cursor*
  - At exactly what position the prevous read stoped
  - We expect further reading
- Receiving pages from PFS
  - Index pages are read to ISAM process and they are processed in ISAM process.
  - Data pages should be directed to TP-AM process (if possible)
- We can have dense/sparse indexes... this should be illuminated
  - (iztok) I suggest that we have **sparse** index
  - The last index level includes `<key,ptr>` where `ptr` points to data page
  - The next `<key,ptr>` points to the next data page
- ISAM index interface
- `qid = open key cp` opens access for a given `key` where `cp` is one of comparison operations equal|eq-more-then|eq-less-then
- `read qid` reads all results of the operation
- `fread qid N` read the next `N` results
- `close qid` closes access to ISAM index
- Batch creation of ISAM index from complete data

Record-based file server (RFS).

- Record file is a sequence of records
- Records are the basic units of operation
- All records have the same length
- Record identifier (RID) identifies a record
- RFS does not know anything about the content of records
- Architecture of RFS
- PFS is an Erlang process
- Records are stored in pages
- RID = sequential number of a record in a file. PID = RID / (num.of records per page).
- RFS interface
- Initialization of the RFS process
- Read N-th record
- Write a new record to the end of file

Triple-pattern access method

- Triple-pattern access method (TP-AM) is a query node
  - TP-AM is a leaf of a query tree
  - TP-AM will be implemented as an update of tp_query_node.erl
  - A description is available at https://big3store.github.io/big3store/
- Architecture of TP-AM
  - TP-AM is an Erlang process
  - Triple-pattern obtained from the message {start ...}
  - TP-AM creates ISAM index process if it does not exist
  - TP-AM opens the access to the ISAM index by providing the access parameters (key+operation)
  - The results of the query run at ISAM index process is obtained via messages from a PFS process
    - Each message contains a data block that is parsed and processed by TP-AM
  - After processing the triples (selection) they are collected in the stream blocks which are sent to the parent query node
  - The protocol of TP-AM is based on
    - the internal state active, db_access, eos, or inactive
    - messages obtained from a PFS process (stored in input queue after receival)
    - state of input and output queues
- TP-AM interface
  - TP-AM accepts synchronous messages
    - {start, QNodeId, QueryId, SessionId, TriplePattern, ...} and
    - {eval, VarsValues}
  - TP-AM accepts asynchronous messages
    - {empty, From} and
    - {stop, From}

Draft material

Interrelations

- PFS is a separate process as defined above
- One ISAM process works with an open PFS process
- The results of an ISAM index access are the leaf pages, more precisely, key/value records in leaf pages.

- In initial version ISAM can read also the results, but it seems that the results can be sent directly to the initiator of ISAM process, i.e., the TP-AM process (triple-pattern access method).