LibreS3: design, challenges, and steps toward reusable libraries
Török Edwin edwin@skylable.com
Skylable Ltd.

Concepts

LibreS3’s architecture is not novel: it is based on existing, well-established concepts. What is interesting is how these all work together in a real application.

Monad with exception handling

type α t (* the type of deferred computations *)
val return : α → α t (* immediate value to deferred value *)
val ( >>= ) : α t → (α → β t) → β t (* chain *)
val fail : exn → α t (* exception to deferred value *)
val try_bind : (unit → α t) → (α → β t) → (exn → β t) → β t (* try_bind m f g) either f or g (*)

Implementable by monadic concurrency libraries: Lwt, Async, but also by the Result module.

Object storage HTTP(S) REST API

Amazon S3 server: proprietary, runs on Amazon’s hardware
S3 server: FOSS cluster storage backend: supports deduplication, no vendor lock-in
LibreS3: FOSS S3-compatible server: Allows to connect existing S3 clients to an S3 cluster.

LibreS3 client

• an HTTP(S) server that implements the S3 REST API on one side
• an S3 client on the other side
• both LibreS3 and S3 run on your own hardware

Libraries

Reusable components of LibreS3, parameterized by a monad: it is not tied to one particular implementation, or to the use of a monadic concurrency library. See also Cohttp, Mirage, ...

any-cache: Least Recently Used cache with 2 Queues.

Implemented as a result monad function

module Make(R:Result) : sig
  type ('ok, 'err) t
  val create : int → ('ok, 'err) t
  val get : ('ok, 'err) t → notfound:'err → string → ('ok, 'err) R.t
  val set : ('ok, 'err) t → string → ('ok, 'err) R.t
  val lookup : ('ok, 'err) t → string → (string → ('ok, 'err) R.t) → ('ok, 'err) R.t
end

monad transformer provided internally

let lift f v = M.try_bind (fun () → f v) return fail

usable with monadic concurrency libraries

val lookup_exn : (a, exn) t → string → (string → a M.t) → a M.t

Testing (same code for Lwt, Async and direct mode)

module Make(Monad: MonadTest) = struct
  (* marked
  Monad.run (lookup cache key1 (compute_data_direct cnt))

any-http

Somewhat similar to how Mirage applications are built

module MyApp(Http:H:HttpInf,S) = struct
  (* ... *)
  let uri = Uri.make --scheme:"http" --host --port path ()
  in
  H.Client.call meth uri (H.Headers.init ())(H.Body.empty)
  >>= fun (status, headers, bodystream) >>= fun body →
                      ( * ... *)
module App = MyApp(Httpservice_cohttp_async)
module App = MyApp(Httpservice_ocsigendserver)
module App = MyApp(Httpservice_cohttp_lwt.Make
                      (Cohttp_lwt_unix.Server) (Cohttp_lwt_unix.Client))

Work in progress

any-to abstract over Unix vs Lwt_unix (Async and Mirage too in the future)
iconvcombiners for describing JOnn/Xmld to OCaml type conversion
ideas from: “Type-safe functional formatted IO” (Oleg Kiselyov), cconv, meta_conv and dynotype. Benefits: don’t have to choose between Yoj
son/Jomn, unified tree/streaming interface and error reporting.

Issues

double-release of runtime lock OCamlNet < 3.7.4, ocaml-sql 0.4.6a
tracked down using a debug patch for OCaml’s runtime and submitted patch
Unix. fork in Ocsigendserver must use Lwt_unix.fork
otherwise it hangs if an error message is printed, patched upstream
ocamlbuild SIGPIPE on *BSD worked around in 4.02
Lwt readdir crash fixed in 2.4.2
OcamNet SSL persistent connections fixed in 3.7.4
PUT/DELETE support in Ocsigendserver implemented in 2.4.0
Ocsigendserver default worker threads 1000 too much, using 64
Mirage XSA-90 / CVE-2014-2580 (crashed Dom0 kernel), not used yet

Packaging

If user has OPAM it is easy, but ...

User may not have “new enough OCaml” RHET/CentOS 6
rebuilt .spec file from Fedora on CentOS6 and provide binary rpm
Libraries requiring newer versions 3.11.2 3.12.1 4.00.1
“portable” Linux binaries 4.0.1-4.1-1s4 and 4.0.1-4.1-1s4 musl+static
opam switches if you really need them
Full source tarball embedded 3rdparty libraries
custom build script, but very slow
Automation lack of a full OASIS/OPAM workflow
Existing oasis2deb, oasis2rpm
Missing (oasis) (opam)2rpm, (oasis) (opam)2deb
Wishlist opam2release (to build full source tarball, .deb, .rpm and “BSD”)

Additional materials

Paper, bibliography, sample code

http://goo.gl/jmF0cn

http://www.skylable.com/tag/libres3/