OCaml Platform v0.1

Anil Madhavapeddy,
Amir Chaudhry, Thomas Gazagnaire,
David Sheets, Philippe Wang, Leo
White, Jeremy Yallop
University of Cambridge and OCamlPro

with a vast amount of help from
OCamlPro, Jane Street, Citrix, and
the wider OCaml community.
What a Platform isn’t

A group of motivated hackers sprint to build a replacement standard library.

![Comic strip showing the proliferation of standards](image-url)
What a Platform isn’t

A group of motivated hackers sprint to build a replacement standard library.

• Hard to get adoption without a domain-specific purpose.

• Tends to be *opinion* based, and fodder for infinite discussion.

• Sustaining maintenance is tough.
What a Platform is!

• **Tooling** that works together beyond just a language, into the full dev lifecycle.

• **Quantitative** metrics to judge if we are succeeding or not.

• **Agility** to judge the impact of changes quickly to keep moving.

Together, these let users judge if the Platform is suitable for *their* needs.
Design Space

Libraries

- Ctypes
- COW
- Omd
Design Space

Libraries
- Ctypes
- COW
- Omd

Tools
- OPAM
- OPAMdoc
- IDE tools
Design Space

Libraries
- Ctypes
- COW
- Omd

Tools
- OPAM
- OPAMdoc
- IDE tools

Resources
- OCaml.org
- Books
- OCamlot
OPAM Progress in 2013

• **OPAM 1.0** released in March 2013

• **OPAM 1.1** beta released Sept 2013
  - Solid bug fixing and improvement released based on *lots* of feedback.
  - Over 100 contributors, 500+ packages, 1500+ unique versions.
  - Migrating to opam.ocaml.org (CC0) as a community-maintained effort.
OPAM contributors growth
OPAM package growth

http://opam.ocamlpro.com
OPAM 1.2 and onwards

- **Windows support** for the tool.
- **Fast compiler switching** (needs relocatable compiler).
- **Binary packages** to share OPAM installations for teaching.
- More **expressive constraint language** for optional dependencies.
- [https://github.com/OCamlPro/opam/issues](https://github.com/OCamlPro/opam/issues)
**Goal:** single source of cross-referenced documentation for all packages.

- **Why it’s hard:**
  - not all packages can be installed simultaneously (solved via OPAM)
  - resolving module inclusion statically leads to huge output sizes.
  - touches almost all parts of the toolstack (compiler, OPAM, build systems)
Codesign: the Platform and compiler are evolving together, not developed in isolation

- **Examples:**
  - Compiler exports itself as a library, which is enough to build custom frontends.
  - `cmt` files expose typed AST (internals) which is enough for IDEs and search tools.
  - `-short-paths` in 4.1 makes long module paths much more usable.
OPAM Doc

- **Typed AST** now written as `cmt` file from 4.00.1 onwards.

- New **bindoc** tool generates `cmd` files which parse ocamldoc comments.

- `cmd` files are separate for multiple translations, tutorials, etc.

- New **opamdoc** tools combines a `cmt` database into a single website with a subset of packages.
Tooling: ocaml.org

- http://ocaml-redesign.github.io/
- http://ocaml-redesign.github.io/pkg/ (both WIP)
Quantitative: Packages

- What are useful metrics for OCaml?
  - **Portability**: OS, CPU arch, compiler version, C bindings, native/bytecode
  - **Maintainer**: responsiveness, documentation coverage, issue URL
  - **Tests**: code coverage, benchmarks
  - **Stability**: interfaces changing a lot?
Quantitative: Packages

- All of these are being built up in the OPAM repository:
  - The *opam* file tracks compiler constraints across 1500 packages.
  - Can statically analyze the archive contents to determine build system.
  - Transitive cones of library coverage ("when Core breaks, does anyone care?")
Let’s evolve together

• We’re building the framework for a standard library tussle that will let us evaluate the fitness of libraries.

• Plan to benchmark and test Core, Batteries, Extlib, Lwt on a variety of platforms and circumstances.

• We make these available on ocaml.org to understand how to achieve consensus. “Why are there so many separate stdlibs?”
The current state

No one candidate is quite supreme yet.

- **Lwt:** very portable, small, quite C heavy, separate module namespace.

- **Batteries:** comprehensive, no syntax extensions, separate namespace, community developed.

- **Core:** hugely comprehensive, weekly releases, poor portability, single namespace, architected at Jane Street.
Open Problem: Build

- None are quite satisfactory yet, and “almost working” ones proliferate.
- Tension between speed of compilation and features and portability.
- Library-based systems sorely needed.
- The goal should be to statically analyze all 1,500 OPAM packages to test hypotheses.
Questions?

- Get involved with OPAM! Particularly documentation + blogs.
- Feedback on redesign to infrastructure:
  
  http://amirchaudhry.com/ocamlorg-request-for-feedback/

platform@lists.ocaml.org