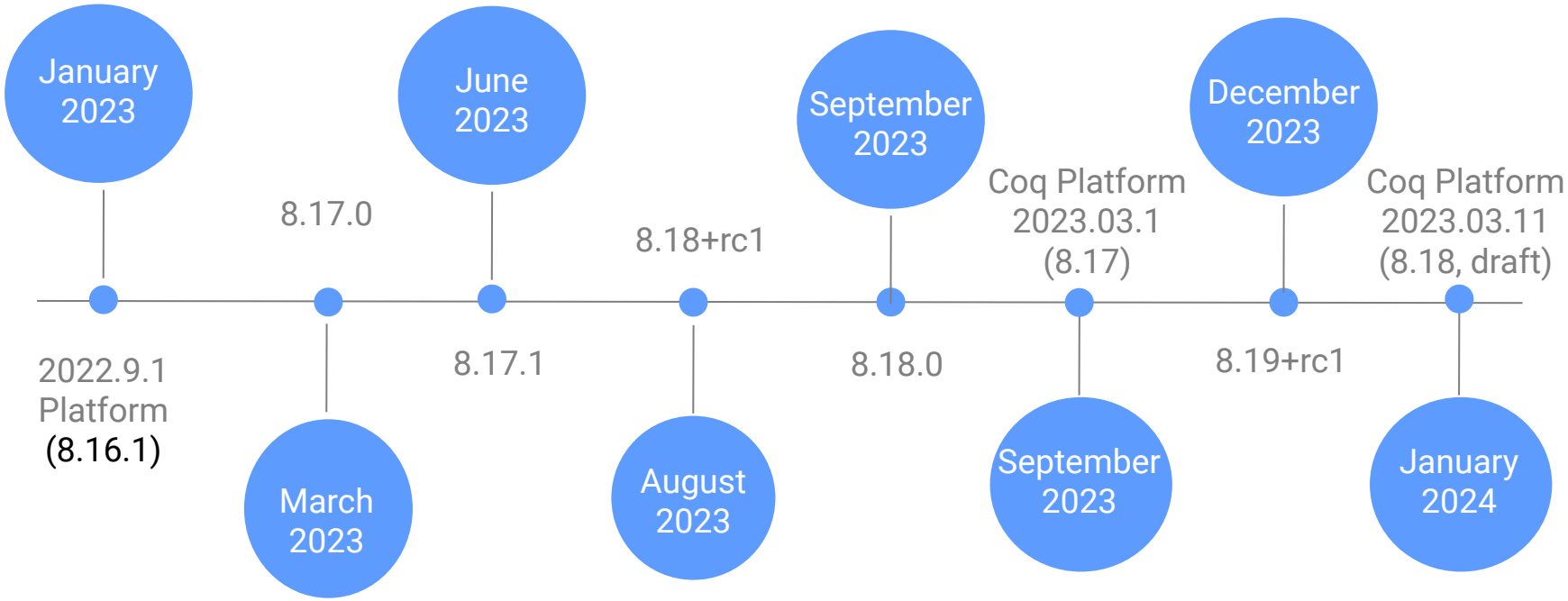
COQ DEVELOPMENT TEAM SESSION

Matthieu Sozeau
CoqPL 2024
London, UK
January 20th 2024

OUTLINE

1. Coq 8.17, 8.18 & 8.19
2. Coq Platform Updates
3. Coq Future & Roadmaps
4. Highlights
5. Q & A

Coq and Platform Release Timeline



Coq 8.17-8.19 Features

https://coq.inria.fr/refman/changes.html

- Notations activation/deactivation
- Temporary scopes, multiple scopes for Arguments
- Ltac2 improvements: richer APIs, case compilation, bugfixes
- Sort polymorphism and unification of sorts.

  Generic definitions over Prop, SProp and Type.
- Stdlib improvements: arithmetic libraries, lists, analysis

Demo

- Sort polymorphism
- Notation activation, selective imports
- Ltac2

Coq 8.17-8.19 Changes

https://coq.inria.fr/refman/changes.html

- Default localities for hints and instances
- Improved control over warnings, providing better support for deprecations. Library deprecation available.
- Lazy, simpl, cbn and eval now can do head reduction
- Precise profiling support
- OCaml 5 compatibility (perf caveats, no `native_compute`)

OUTLINE

1. Coq 8.17, 8.18 & 8.19
2. **Coq Platform Updates**
3. Coq Future & Roadmap(s)
4. Highlights
5. Q & A

Coq Platform

A coherent *distribution* of Coq packages.

Main objectives: easy, standard, tested

- Installers for Windows, Linux (snap), OS X
- Customizable! Just choose a package list (e.g. for lectures)

Coq Platform Charter

Maintenance: Michael Soegtrop & Romain Tetley

Editorial board: Reynald Affeldt, Andrew Appel, Yves Bertot,

Michael Soegtrop & Matthieu Sozeau

Coq Platform 2023.3.1 (8.17)

OUTLINE

1. Coq 8.17, 8.18 & 8.19
2. Coq Platform Updates
3. **Coq Future & Roadmaps**
4. Highlights
5. Q & A

In the pipeline

- Add rewrite rules to Coq (Y. Leray, T. Winterhalter, G. Gilbert, …)
- Sort polymorphism use in the stdlib (P.M. Pédrot, G. Gilbert)
- Algebraic universes for all (PR#16022, M. Sozeau, M. Bezem)
- Ltac2 maturation as a replacement for Ltac1 (P.M. Pédrot, G. Gilbert)
- Verified Extraction integration

See the short-term roadmap for Coq (CEP#69)

Renaming

The Coq Proof Assistant will be renamed The Rocq Interactive Theorem Prover (abbreviated, The Rocq Prover or simply Rocq).

- User-survey results: split on the renaming, majority for renaming or neutral Development team majority for the renaming.
- Pronounced /ɹɑk/. Pays homage to Rocquencourt (birthplace of Coq), suggestive of rock-solid software.
- Plan: make the renaming effective this year, with an updated visual identity, website and first release. In the meantime, keep using Coq!
- We are aware of the Roc programming language and the converse is true as well. We believe it will be easy to distinguish them: they have very different use cases, and one can use Rocq Prover/ITP in case of ambiguity.

Long-term vision

The Rocq Interactive Theorem Prover will be **collaboratively** developed with a focus on:

Genericity: integrating various logical "frameworks" in a single system:

- Building on sort polymorphism, rewrite rules and metalanguages
- New sorts for effectful computation (à la BTT, Pédrot & Tabareau), erasable data (Keller & Lasson), …
- Integration of observational type theory (UIP, FunExt, Quotient types)
- Enabling efficient embedded domain-specific logics (à la Iris)
- Provide bridges between developments: transfer tools

With continued support for the wide variety of developments we have today

Long-term vision

Robustness: a platform for high-assurance software production.

High-assurance and high-performance implementations of the kernel and extraction/compilation facilities, minimizing the TCB, relying on **formal** metatheory.

Accessibility and productivity: bringing Rocq to students and engineers.

Intensive efforts to provide better user interfaces based on off-the-shelf development environments, see talks by Romain and Emilio this afternoon!

Strong automation tools and library management support. We need you!

Planned Events

- Coq Users and Developers Workshop 2024
- Coq Workshop 2024 in Tbilisi, Georgia

OUTLINE

1. Coq 8.17, 8.18 & 8.19
2. Coq Platform Updates
3. Coq Roadmap & Future
4. **Highlights**
5. Q & A

Highlight: Coq-Elpi & applications

Coq-Elpi: high-level λProlog metalanguage for Coq (tutorial) (Enrico Tassi)

Hierarchy-builder: structure hierarchies à la math-comp from declarative specifications (Cyril Cohen & al)

Trakt: an extensible framework for transfer between (non-isomorphic) theories (ERC Fresco, Enzo Crance PhD)

Liber Abaci: revisiting elaboration phases for natural mathematical expression (e.g. fine tuning Type classes and canonical structures).
Inria project led by Yves Bertot.

Highlight: MetaCoq, CertiCoq, ConCert

Verified erasing compilation pipelines from **Gallina** through **λ□** to:

★ **C** compilable by CompCert/clang/gcc

CertiCoq: bootstrappable, with a verified GC! (Appel et al)

VeriFFI: link with VST code (Korkut, Stark & Appel)

CertiCoq-Wasm: alternative code generator (Meier et al)

★ **Malfunction** / OCaml: with a restricted, safe .mli interface

(Forster, Sozeau & Tabareau)

★ **Web** / Smart Contract Languages (Liquidity, Elm/MidLang)

ConCert (Annenkov et al). Uses a *type erasure* phase

Q & A Time!