Coq Coding Sprint

Enrico Tassi – 22/6/2015

Ok, ok, we got the name wrong ;-)

1. This is not a Coding Sprint, it is an Hackaton!
2. In our defense:
   - It is the first one, no idea you would be so many!
   - THANKS for coming!
3. Who cares about the name?!

Why are we here?

1. Have fun making Coq a better project
2. Learn something new about Coq
   - but it is not a traditional “school”
   - read: you will work harder than the “teachers”

Roadmap of this talk

1. General info (food, patches...)
2. Bird eye view of Coq and its internals
3. Demo: writing a simple plugin

1

General infos

Program

<table>
<tr><th></th><th>Monday 22</th><th>Tuesday 23</th><th>Wednesday 24</th><th>Thursday 25</th><th>Friday 27</th></tr>
<tr><td>8 AM</td><td></td><td rowspan="2">room (7)</td><td rowspan="2">room (7)</td><td rowspan="2">room (7)</td><td></td></tr>
<tr><td>9 AM</td><td rowspan="3">Arrival (1)</td><td rowspan="4">Coq WS (8)</td></tr>
<tr><td>10 AM</td><td>Code (5)</td><td>Code (5)</td><td>Code (5)</td></tr>
<tr><td>11 AM</td><td></td><td></td><td></td><td></td></tr>
<tr><td>12 PM</td><td colspan="4">Lunch</td></tr>
<tr><td>1 PM</td><td colspan="4"></td></tr>
<tr><td>2 PM</td><td rowspan="2">Intro (2)</td><td rowspan="4">Code (5)</td><td rowspan="4">Code (5)</td><td>Code (5)</td></tr>
<tr><td>3 PM</td><td rowspan="2">Debriefing (6)</td></tr>
<tr><td>4 PM</td><td rowspan="2">Round table (3)</td></tr>
<tr><td>5 PM</td></tr>
<tr><td>6 PM</td><td rowspan="2">Pub (4)</td><td>room (7)</td><td>room (7)</td><td>room (7)</td></tr>
<tr><td>7 PM</td><td></td><td></td><td></td><td></td></tr>
</table>

Food: 2 options

1. Here at Inria, nothing fancy but good price: 7.50€
2. St. Philippe (700m downhill) offers more choice but at higher prices. Eg. chees burger with fries or dish of the day at ~ 15€.

Contributions: where/how

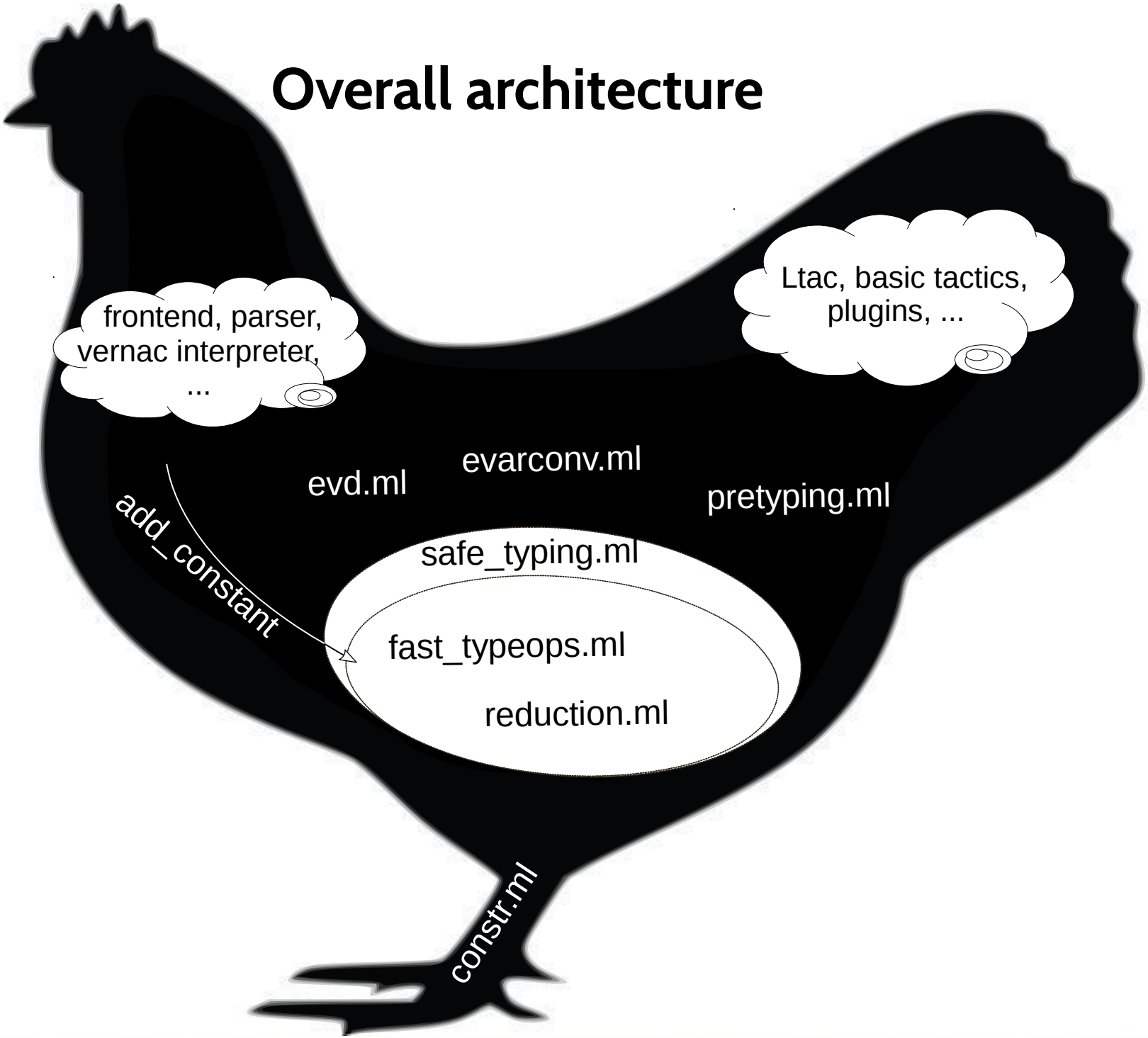
1. Plugin: put your code on github
2. Patches: pull requests to coq/coq
3. Bugfix: coordinate using the list of "simple" bugs page and the bugtracker

In all cases, please log your activity in the dedicated page

2

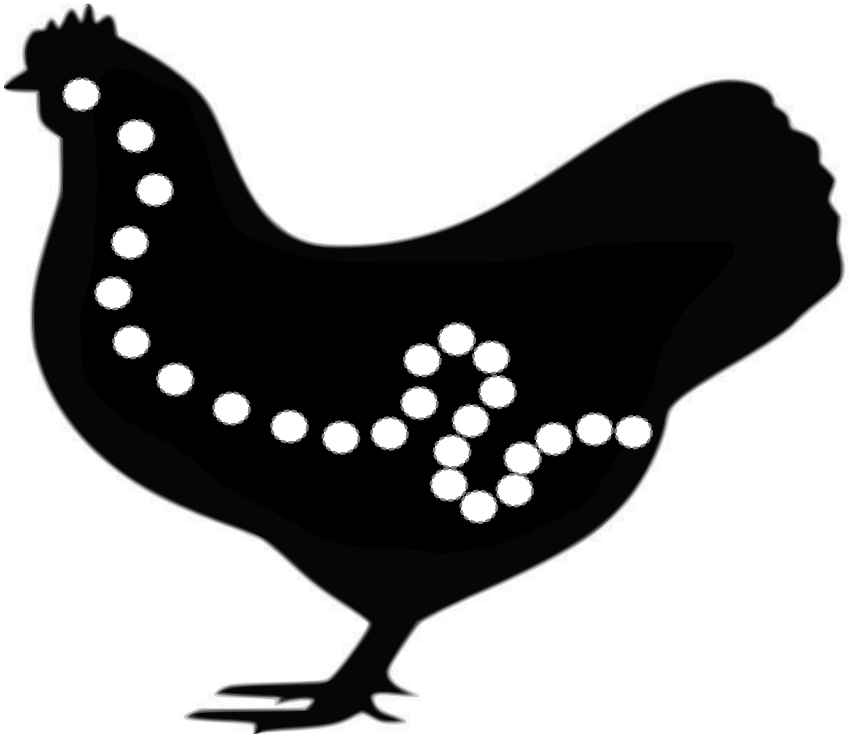
Bird eye view of Coq's internals

Overall architecture



Guided tour

```
Definition foo :=
  fun x => x = 3.
Print foo.
```



```
foo : fun x : nat => x = 3
```

Data types and transformations

string

```
Definition foo :=
  fun x => x = 3.
Print foo.
```

parsing

constr_expr (AST)

```
VernacDefinition( “foo”,
  DefinedBody(
    CLambdaN([ “x”, CHole],
      CNotation( “_ = _”,
        [ CRef “x”; CPrim 3 ])))
VernacPrint (PrintName “foo”)
```

internalization
(notations, globals, implicit args)

glob_constr (untyped)

```
GLambda( “x”, GHole,
  GApp( GRef “Coq.Init.Logic.eq”,
    [ GHole;
      GVar “x”;
      GApp( GRef “Coq.Init.Datatypes.S”,
        […GRef “Coq.Init.Datatypes.O”...])]))
```

pretyping
(De Bruijn idxs,coercions,…)

constr (typed)

```
Lambda( “x”, Ind “Coq.Init.Datatypes.nat”,
  App( Ind “Coq.Init.Logic.eq”,
    [ Ind “Coq.Init.Datatypes.nat”;
      Rel 1;
      App(Construct “Coq.Init.Datatypes.S”,
        […Construct “Coq.Init.Datatypes.O”...])]))
```

Data types involved

string

```
fun x : nat => x = 3.
```

printing

constr_expr

```
CLambdaN([ “x”, CRef “nat” ],
  CNotation( “_ = _”,
    [ CRef “x”; CPrim 3 ])))
```

externalization

glob_constr

```
GLambda( “x”, GRef “Coq.Init.Datatype.nat”,
  GApp( GRef “Coq.Init.Logic.eq”,
     [ GRef “Coq.Init.Datatypes.nat”;
       GVar “x”;
       GApp( GRef “Coq.Init.Datatypes.S”,
         […GRef “Coq.Init.Datatypes.O”...])]))
```

detyping

constr

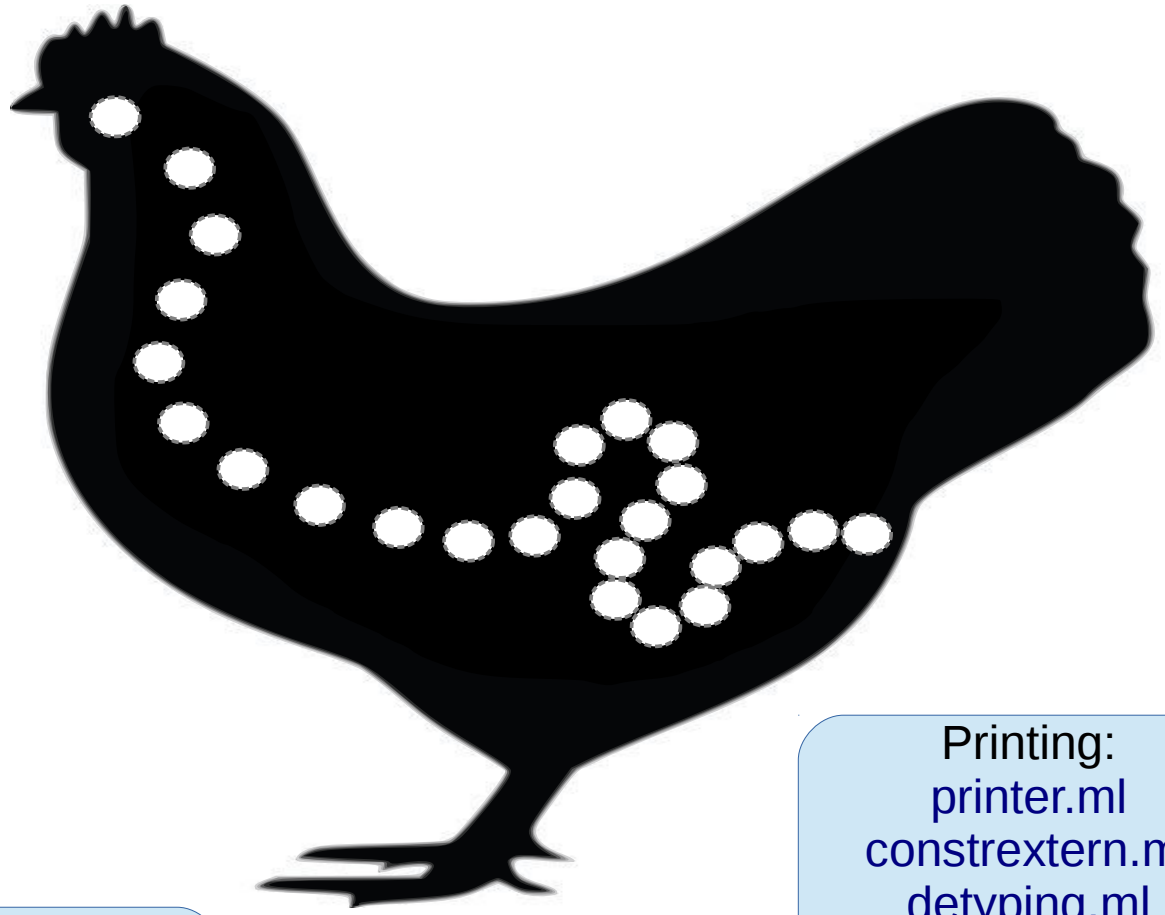
```
Lambda( “x”, Ind “Coq.Init.Datatypes.nat”,
  App( Ind “Coq.Init.Logic.eq”,
     [ Ind “Coq.Init.Datatypes.nat”;
       Rel 1;
       App( Construct “Coq.Init.Datatypes.S”,
         […Construct “Coq.Init.Datatypes.O”...])]))
```

Where's the code?

Frontend:
vernac.ml

Parsing:
g_vernac.ml4 g_constr.ml4
vernac_expr constr_expr

Interpreter:
vernacentries.ml
(dumbglob.ml)



Printing:
printer.ml
constrextern.ml
detyping.ml

Term internalization:
constrintern.ml
notation.ml
glob_constr

Type inference:
pretyping.ml
constr

3

Demo

Demo

1. code is at github/gares/example_plugin
2. adds 1 tactic (intro) coded in "Curry-Howard style"
3. adds 1 vernacular (print)

4

Next

Roundtable

1. Everybody with a project in mind talks about it (5' max)
2. So that we know what you are going to do
3. So that you can group with others working on similar projects

If you are looking for an idea, here there are some