TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Canon

CWI Centrum Wiskunde & Informatica

Bacatá: Notebooks for DSLs, Almost for Free

Mauricio Verano Merino, Jurgen Vinju, and Tijs van der Storm.

<Programming> March 2022.

Vol. 4

What is a “Notebook”

Static output

Rich media output

<Programming 2021>

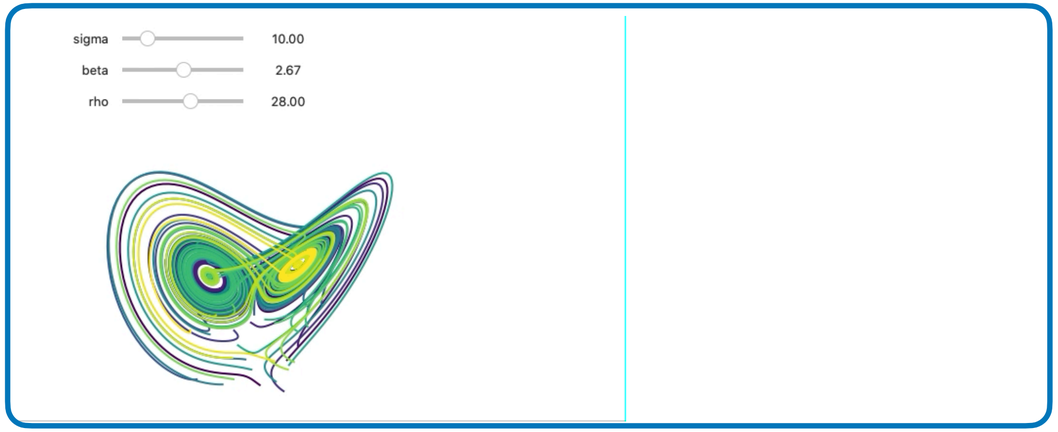
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus tempus hendrerit lacus, sed tempor leo tristique et. Curabitur sit amet vulputate est.

[1]
```
for index, row in raw_data.iterrows():
    countries[row['Country']] = countries.get(row, 0) + 1
```

[1]
```
Index(['pop_est', 'continent', 'Country', 'iso_a3'], dtype='object')
```

[2]
```
from lorenz import solve_lorenz
w=interactive(solve_lorenz,sigma=(0.0,50.0),rho=(0.0,50.0))
```

[2]



Documentation cell

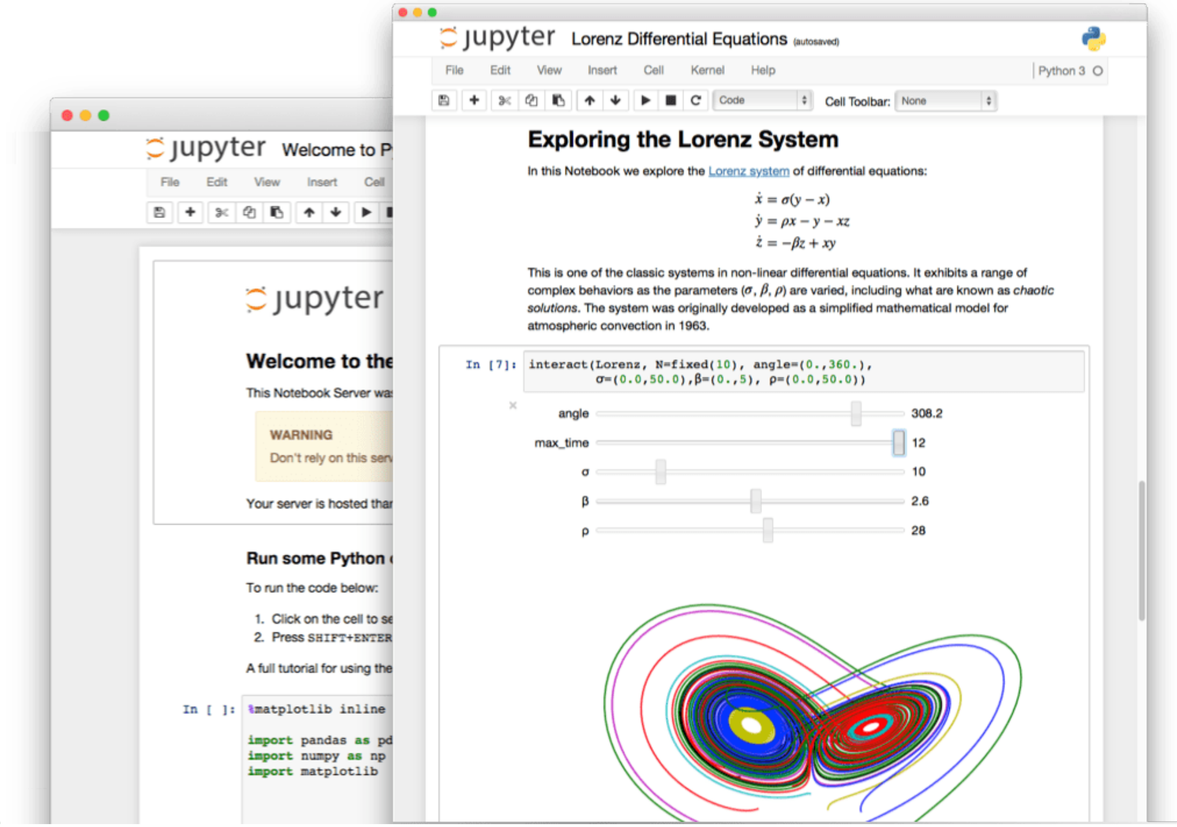
Code Input cell

Value Output cell

Jupyter Notebook

http://www.jupyter.org

- Platform for computational narratives
  - Live code
  - Equations
  - Narrative text
  - Interactive user interfaces
  - Reproducibility

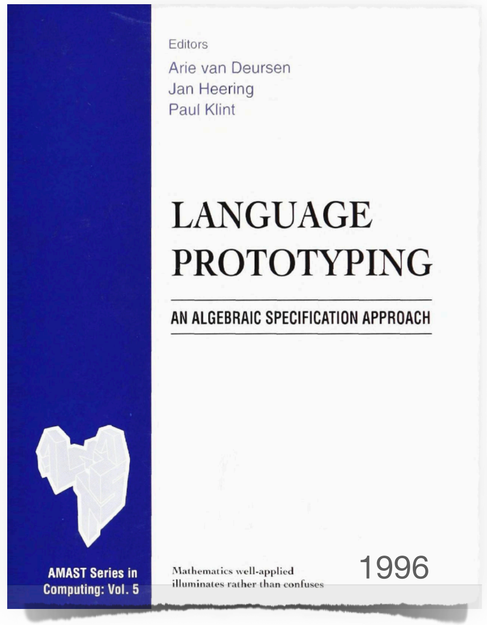


There are **millions of notebook users**
[Pimentel '19, MSR]

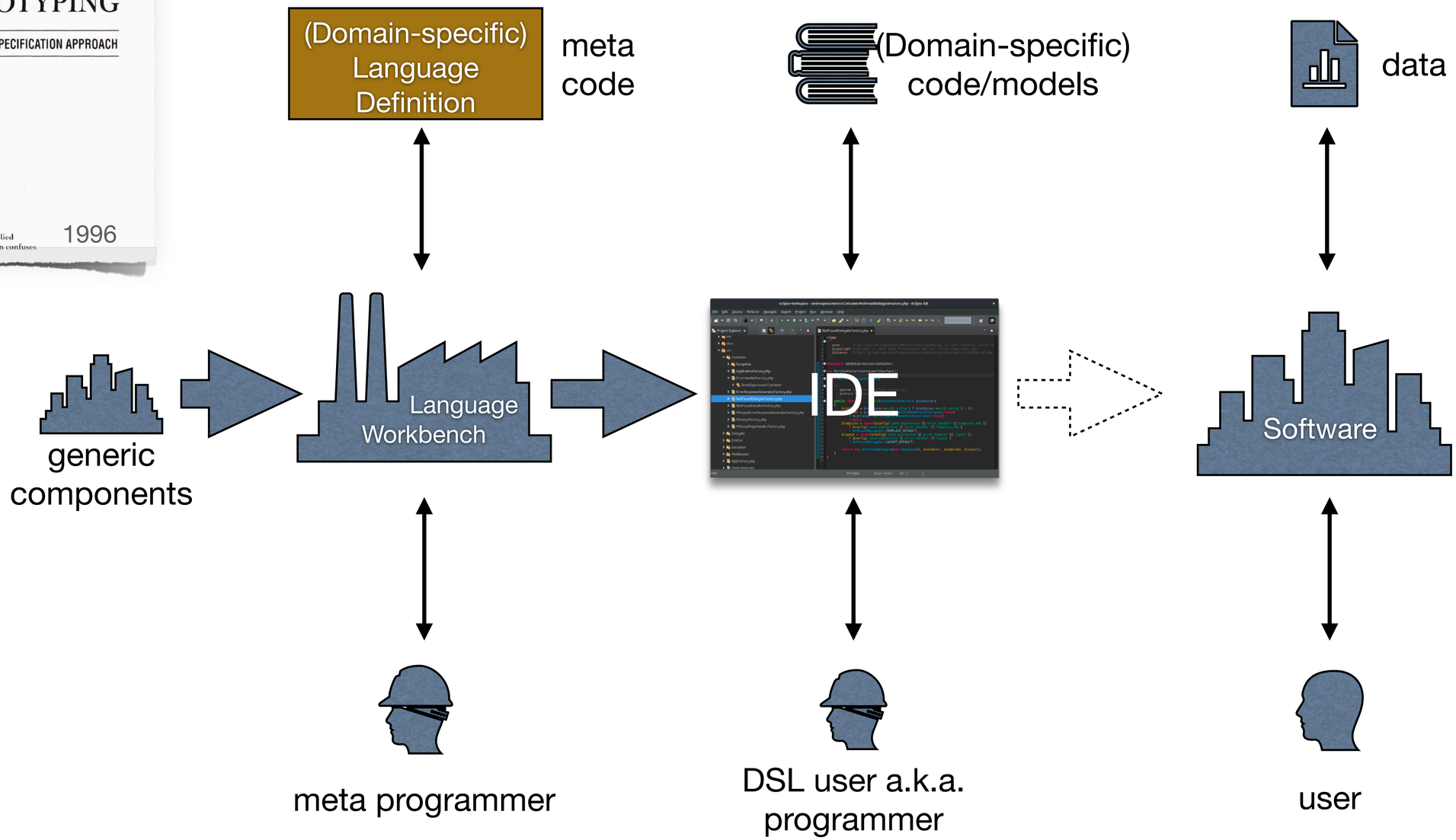
Domain Specific Languages

motivated and problematized

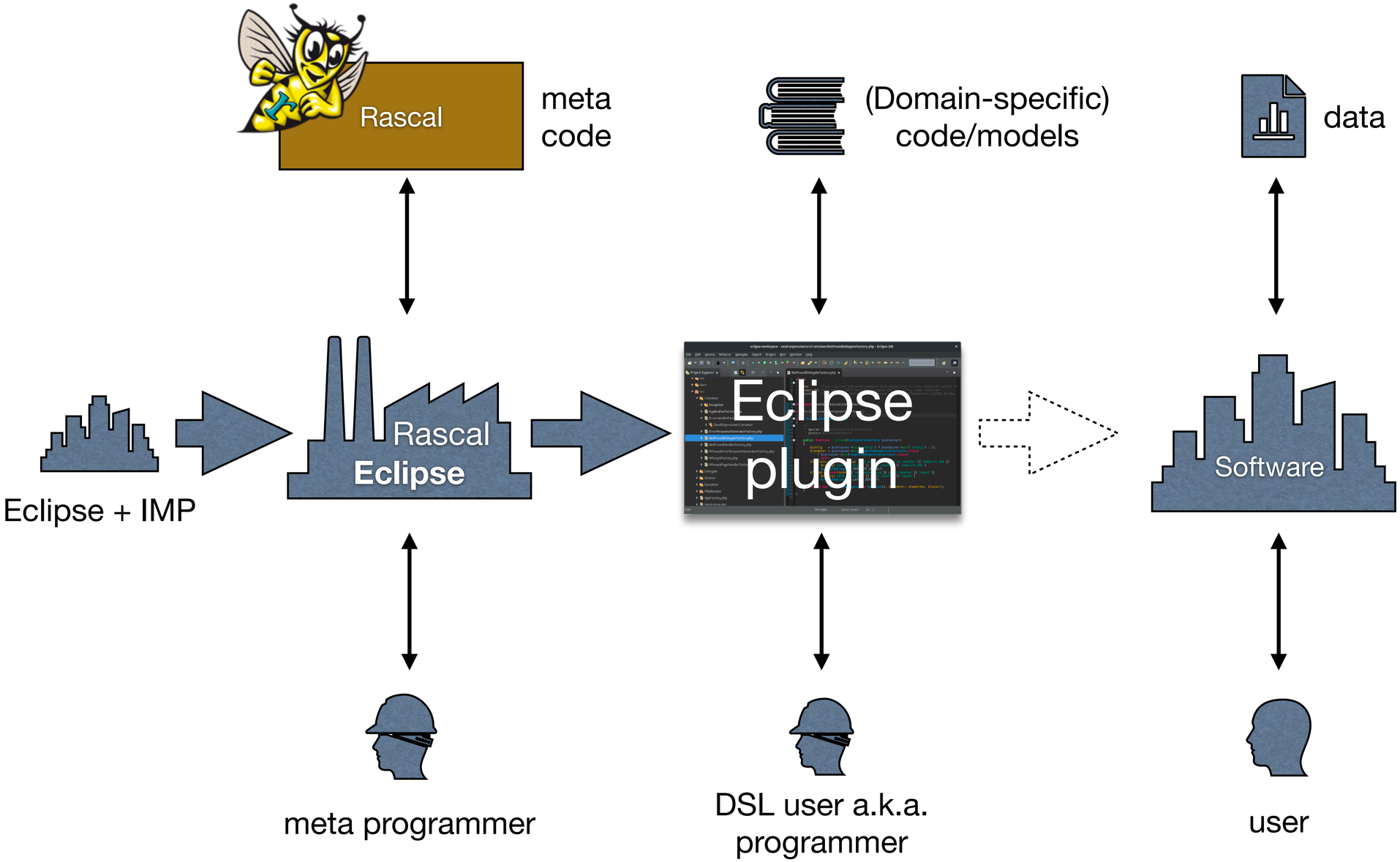
- DSLs are tools for improved **communication** and **knowledge** management
- **Notebooks** could improve **DSL usability & learnability** beyond IDEs
- But, **language engineering** comes with **cost** and **risk**
- So, how can we efficiently construct and evolve notebooks for DSLs?



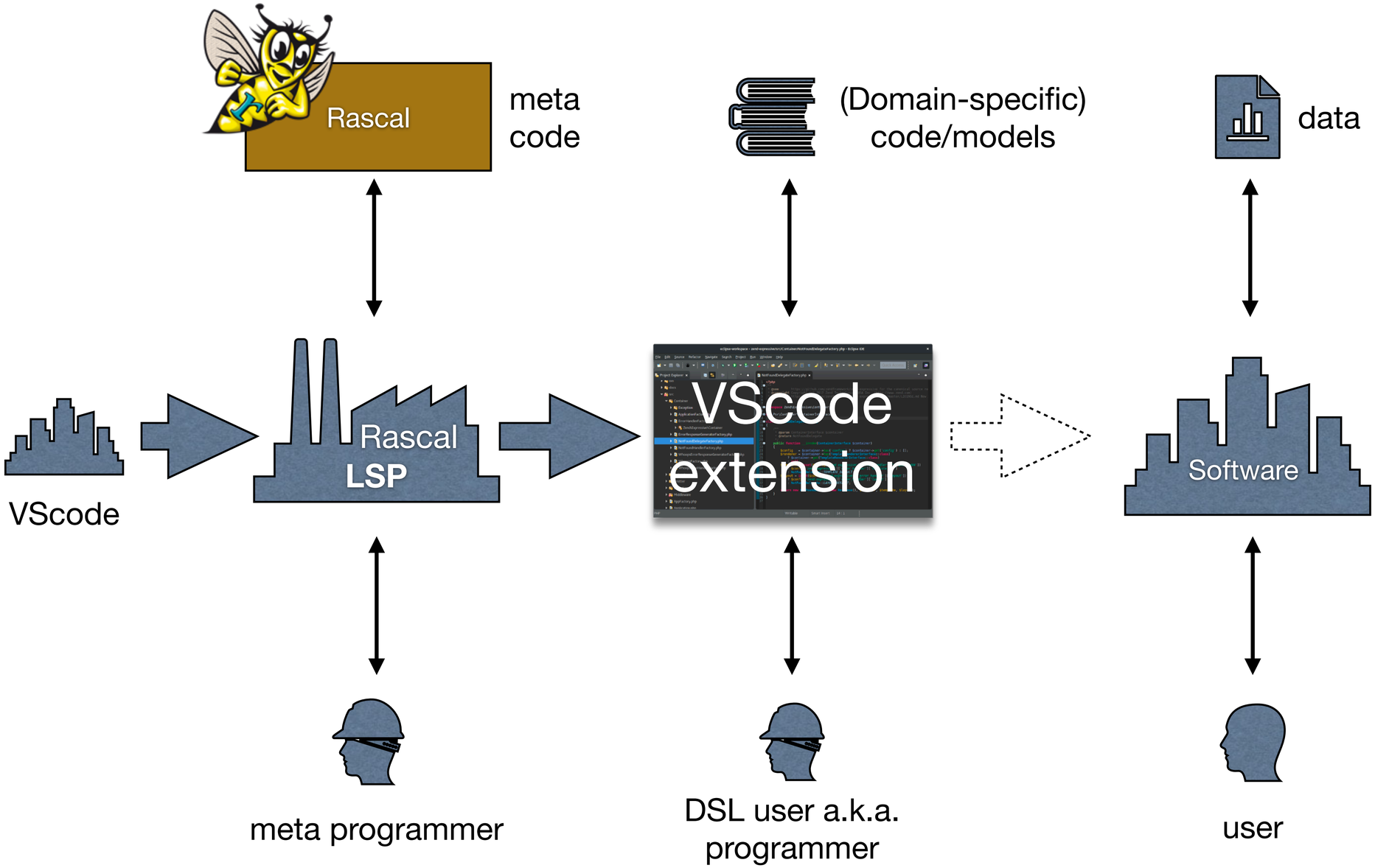
Generating IDEs



Generating IDEs



Generating IDEs



Generating Notebooks

meta code

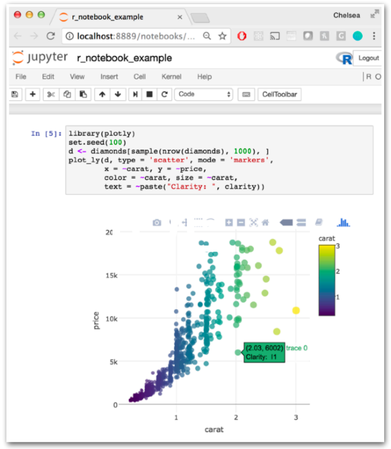
(Domain-specific) code/models

data

jupyter

+ Bacatá-core

Bacatá-Rascal



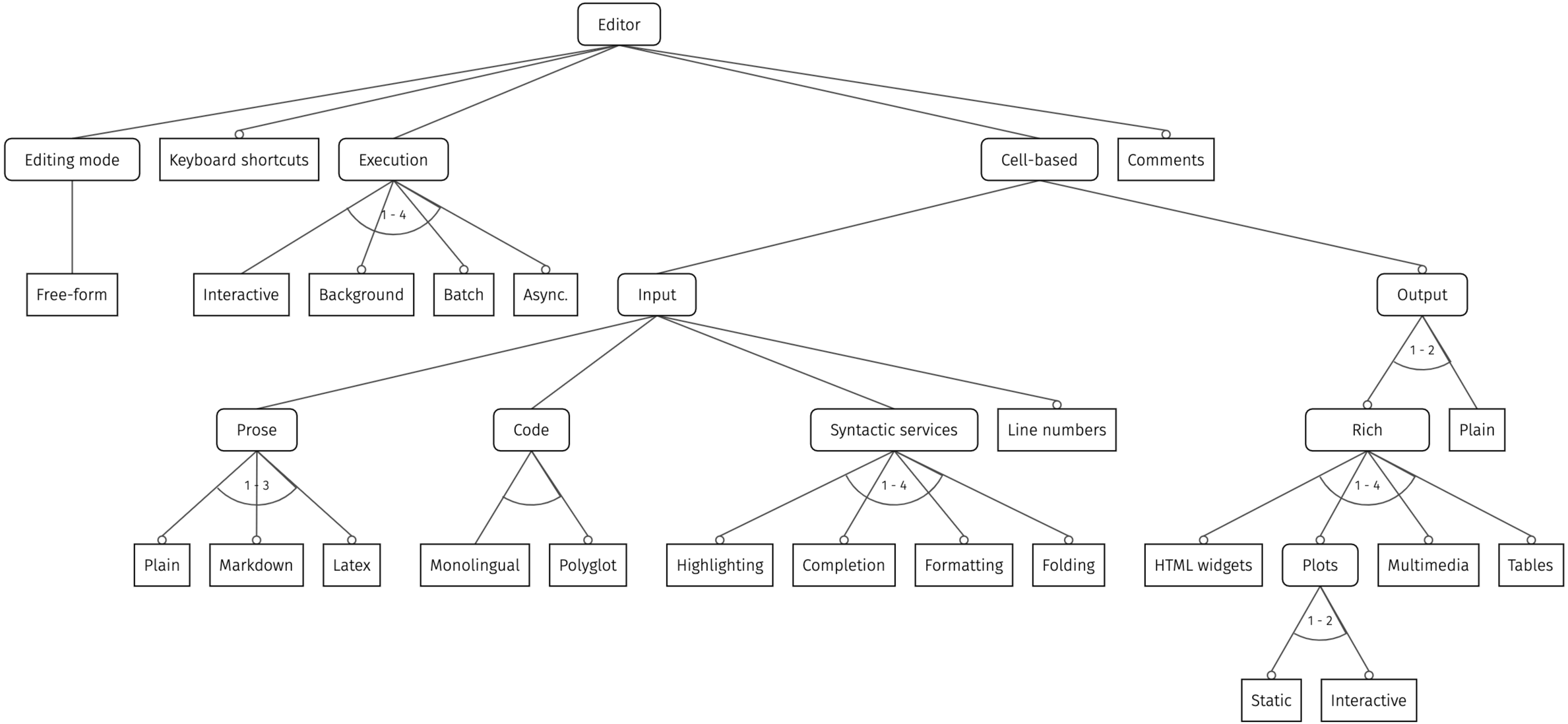
Software

meta programmer

DSL user a.k.a. programmer

user

Domain analysis: computational notebooks



Goal: define and generate notebooks at language abstraction level, and not at the tool implementation level.

```
eval(`<Exp e1> + <Exp e2>`)
     = eval(e1) + eval(e2);
```

(definitional interpreter)

```
public void startServer() throws
    JsonSyntaxException,
    JsonIOException,
    FileNotFoundException,
    RuntimeException,
    UnsupportedEncodingException {
            try (ZContext context = new ZContext(2)) {
                comms = new Communication(connection, context);
            …
            }
}
```

(sockets, async, exceptions, JSON, …)

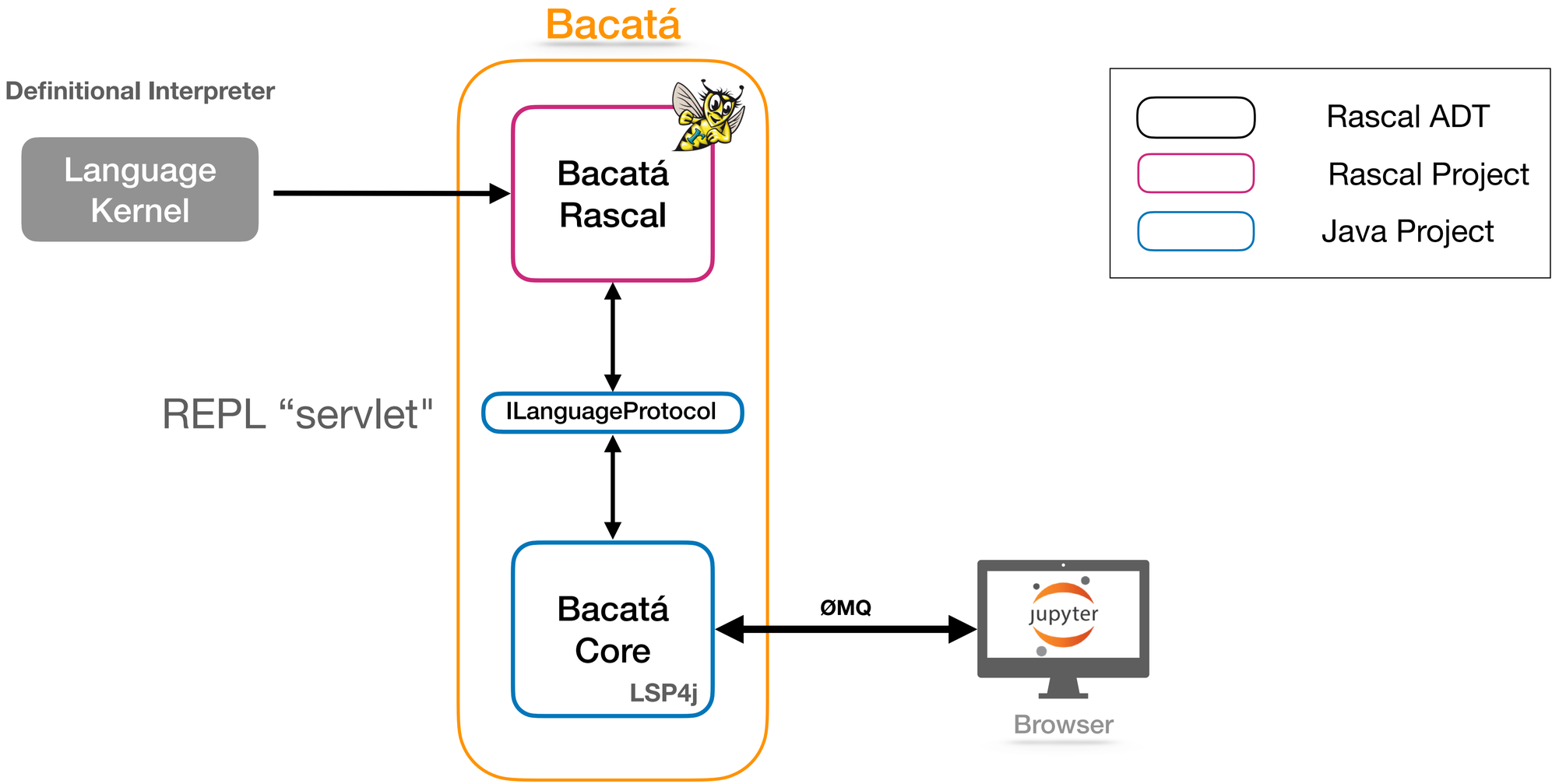
Bacatá

A language parametric notebook generator for DSLs written using Rascal language workbench, which reuses language components.

Objectives:

- Offer the interactive notebook metaphor for DSLs
- Extend the set of generated IDE services of language workbenches.
- Generate DSL notebooks with minimum effort.

Bacatá's Architecture

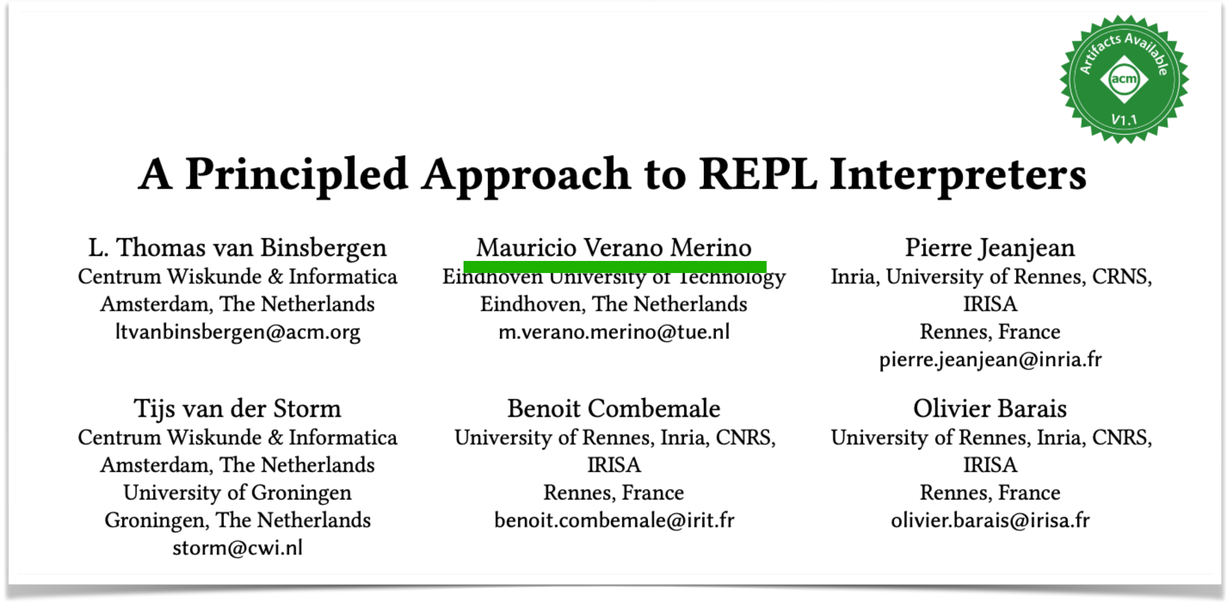


Eclipse LSP4j - third-party library for LSP servers in Java

Intermezzo: REPLs

read-eval-print-loops

$$[\![p_1 \fatsemi p_2]\!] = [\![p_2]\!] \circ [\![p_1]\!]$$



Onward! @ SPLASH 2022

- A “sequential language” is a language that features **associative composition** of programs
- Sequential languages map easily to a “REPL” because REPL interaction reflects sequential program input
- Non-sequential languages can be extended to have a “;” operator

Creating a Notebook

- Define a REPL

```
data REPL
  = repl(Result(str) interpret, Completion(str) complete);

data Result
  = plain(str result, list[Message] messages) // output
  | salix(SalixApp[&T] salixApp);  // interactive webapp
```

- Create a Kernel

```
data Kernel
  = kernel(str languageName, loc project,
           str replFunction, loc logo = |tmp:///|);
```

- Create a Notebook

```
rascal> k = kernel(…);
rascal> nb = bacata(k);
rascal> nb.serve();
The notebook is running at: |http://localhost/…|
```

Demo: Calc Language

Syntax

```
start syntax Cmd
    = Id "=" Exp
    | "show" Exp
    | Exp;

syntax Exp
    = left Exp "*" Exp
    > left Exp "+" Exp
    | Num
    | Id;
```

REPL

```
REPL calcRepl() {
  Env env = ();

  Response calcHandler(str input) {

   Cmd cmd = parse(#start[Cmd], input);

   if ((Cmd)`show <Exp e>` := cmd) {
      return salix(expApp(e, env));
   } else {
      <env, n> = exec(cmd, env);
      return plain("<n>");
   }
  }
  return repl(calcHandler);
}
```

Notebook

```
void calcNotebook() {
   Kernel k = kernel("Calc", |home:///calc/src|,
                     "CalcREPL::calcREPL");
   NotebookServer nb = bacata(k);
   nb.serve();
}
```

jupyter Untitled Last Checkpoint: a few seconds ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Trusted Calc

Run Code

In [ ]:

Case Studies

In [16]:
```
// Select query demo
var myList = [{Name:"Chris",Surname:"Bell"},
              {Name:"Joe",Surname:"Ross"},];

var q = select Name from myList where Name === "Chris";

console.log("Query output: ");
console.log(q);
```

Out[16]:

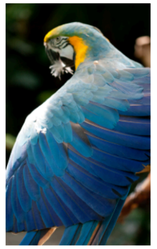
Desugared JS source | Console output

```
// Select query demo
var myList = [{Name:"Chris",Surname:"Bell"},
              {Name:"Joe",Surname:"Ross"},];

var q = JSLINQ(myList)
  .Where(function(item) { return item.Name === "Chris"; })
  .Select(function (item) { return {Name: item.Name}; });

console.log("Query output: ");
console.log(q);
```

SweeterJS

In [1]:
```
Halide::Buffer<float> in = load_and_convert_image("rgb.png");
```

Out[1]:



In [2]:
```
Halide::Func blur(Halide::Buffer<float> in){
    float sigma = 1.5f;
    Var x,y,c;

    Func kernel;
    kernel(x) = exp(-x*x/(2*sigma*sigma))/(sqrtf(2*M_PI)*sigma);

    Func in_bounded;
    in_bounded = BoundaryConditions::repeat_edge(in);

    Func blur_y;
    blur_y(x,y,c) = (kernel(0)*in_bounded(x,y,c)+kernel(1)
                    *(in_bounded(x,y-1,c) + in_bounded(x,y+1,c))+
                    kernel(2)*(in_bounded(x,y-2,c)+
                        in_bounded(x,y+2,c))+
                    kernel(3)*(in_bounded(x,y-3,c)+
                        in_bounded(x,y+3,c)));

    Func blur_x;
    blur_x(x,y,c) = (kernel(0)*blur_y(x,y,c) +
                    kernel(1)*(blur_y(x-1,y,c)+blur_y(x+1,y,c))+
                    kernel(2)*(blur_y(x-2,y,c)+blur_y(x+2,y,c))+
                    kernel(3)*(blur_y(x-3,y,c)+blur_y(x+3,y,c)));

    blur_y.compute_root();

    return blur_x;
}
```

In [3]:
```
Halide::Buffer<float> output1 = blur(in)
        .realize(in.width(), in.height(), in.channels(), ".png");
```

Out[3]:

Loop nests | Execution metrics | Lowered code | Assembly code

C code | LLVM assembly code

```
produce blur_y:
              for c:
                for y:
                  for x:
                    blur_y(...) = ...
            consume blur_y:
              produce blur_x:
                for c:
                  for y:
                    for x:
                      blur_x(...) = ...
```

Halide*

In [1]:
```
form myForm = taxOfficeExample {
  "Did you buy a house in 2010?"
    hasBoughtHouse: boolean

  "Did you enter a loan?"
    hasMaintLoan: boolean

  "Did you sell a house in 2010?"
    hasSoldHouse: boolean

  if (hasSoldHouse) {
    "What was the selling price?"
      sellingPrice: money
    "Private debts for the sold house:"
      privateDebt: money
    "Value residue:"
      valueResidue: money = sellingPrice
          - privateDebt
  }
}
```

Out[1]: ok

In [2]:
```
html(myForm)
```

Out[2]:

Form: taxOfficeExample

Did you buy a house in 2010? true false

Did you enter a loan? true false

Did you sell a house in 2010? true false

What was the selling price? 243,000.00

Private debts for the sold house: 39,250.00

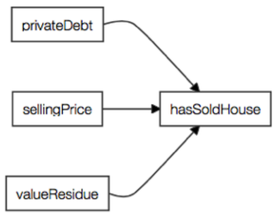
Value residue: 203,750.00

Submit

In [3]:
```
visualize(myForm)
```

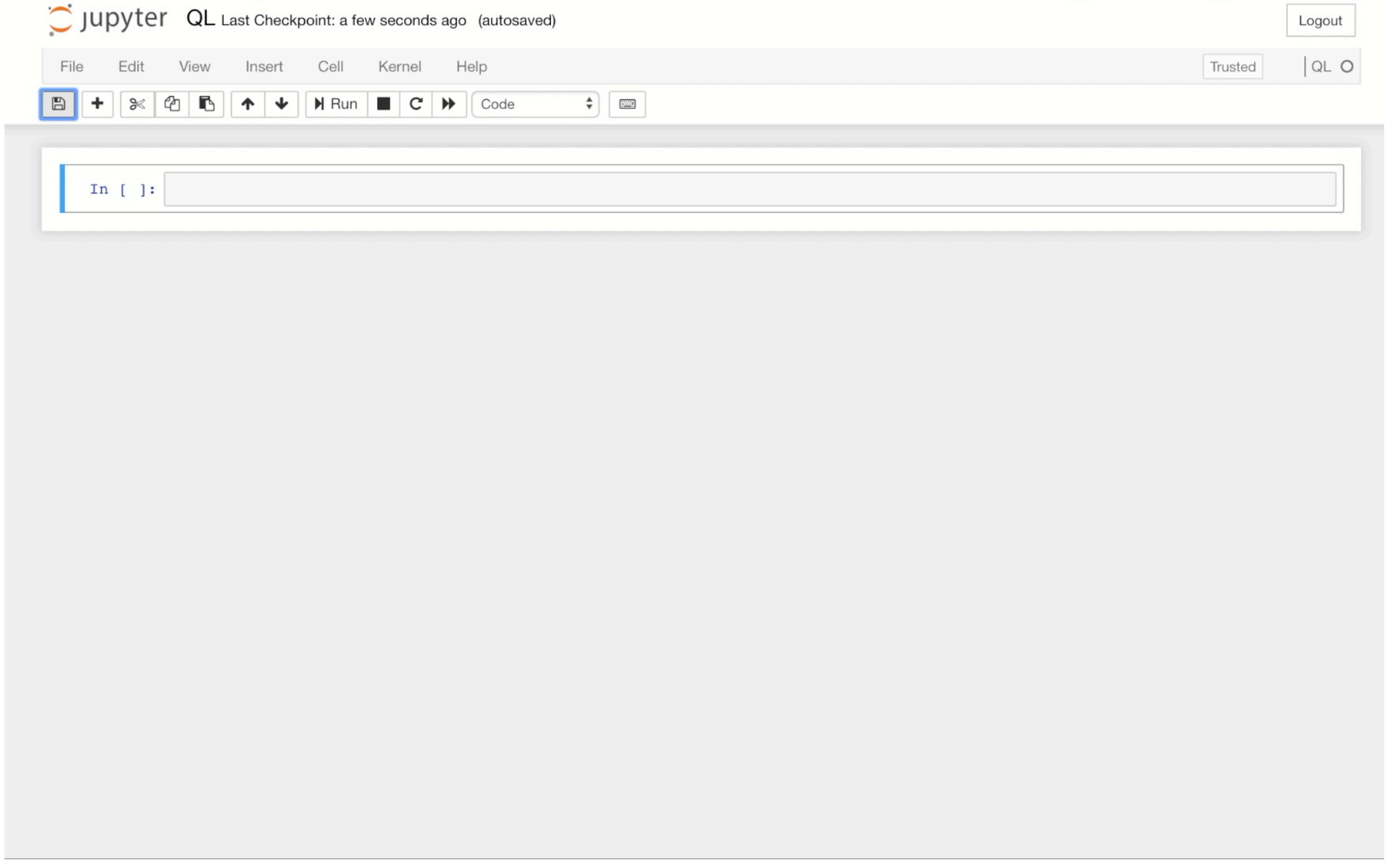
Out[3]:

Visualization

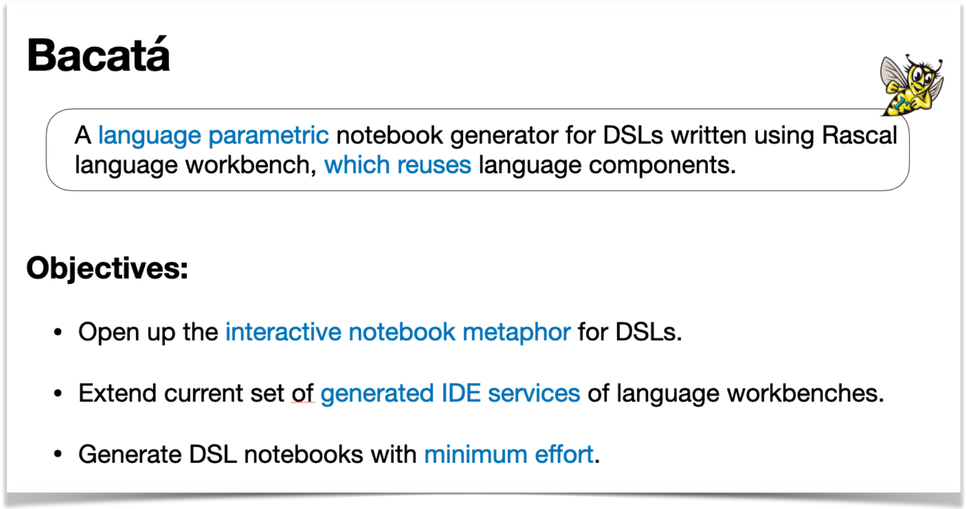


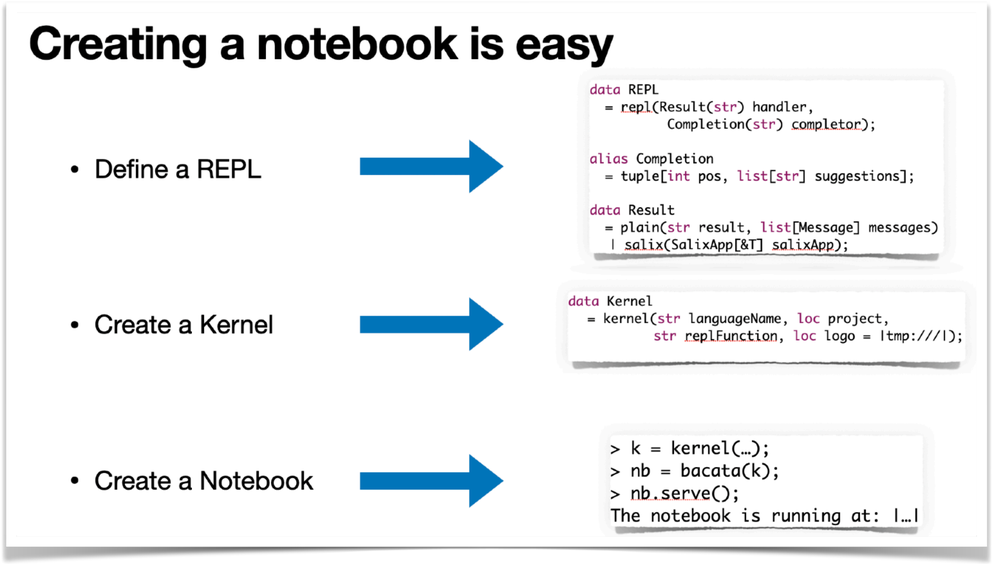
Questionnaire Language (QL)

Case Study - Questionnaire Language



Bacatá: Notebooks for DSLs, Almost for Free





- Notebooks are an interesting design point
- DSLs and Notebooks; a good match!
- **Bacatá: language-parametrized notebook**
  - generates Jupyter kernels for DSLs;
  - **principled REPLs** [Onward! 2020]
  - **“Yet another” IDE generator**
  - Part of the Rascal ecosystem

Mauricio Verino Merano @ VU Amsterdam

http://www.rascal-mpl.org