Case study

Doing web-based data analytics with F#

Tomas Petricek
University of Cambridge

Don Syme
Microsoft Research, Cambridge

Get in touch: tomas@tomasp.net | @tomaspetricek | @dsyme

‹rant›

According to [the] proponents [of new experimentalism], experiment can have a "life of its own" independent of a large-scale theory.

Ian Hacking (1983)
Representing and Intervening

‹/rant›

‹rant›

Relevant **case study**

Look at non-trivial real-world problem

Theory or **language independent**

Produce significant visible result

Combination of language features

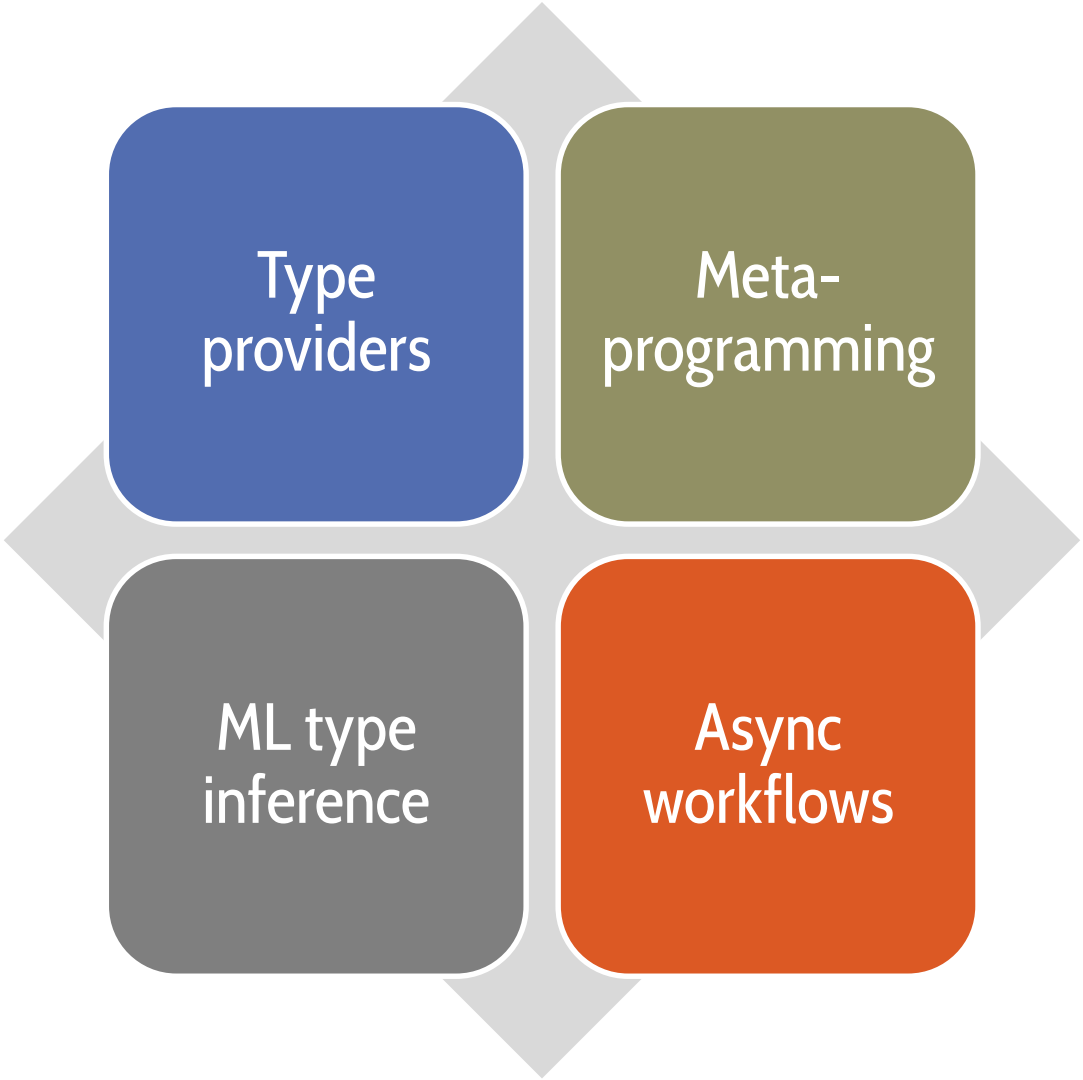
Arising from the ML tradition

‹/rant›

Demo

Doing web-based data analytics with F#

How does this work?



Meta-programming

Meta-programming

Light-weight meta-programming

Pick one aspect and do it well

Heterogeneous execution (CUDA, SQL, JS, …)

Implicit and explicit quotations

```
[<ReflectedDefinition>]
module Program
```

D. Syme. *Leveraging .NET meta-programming components from F#*, ML workshop 2006

JavaScript integration

Meta-programming

F# to JavaScript translation

F# **semantics** or JavaScript **semantics**?

F# **libraries** or JavaScript **libraries**?

TypeScript type provider

```
type j = TypeScript.Api<"jquery.d.ts">
type h = TypeScript.Api<"highcharts.d.ts">
```

T. Petricek. *Client-side scripting using meta-programming.* BSc thesis. Charles University, 2007

Asynchronous workflows

Async workflows

Single-threaded semantics

Close to F# GUI threading model

Syntax matters!

```
let render () = async {
  let opts = h.HighchartsOptions()
  for country, check in infos do
    let! data = country.Indicators.
      ``School enrollment (% gross)`` (* ... *) }
```

D. Syme, T. Petricek, D. Lomov, *The F# Asynchronous Programming Model*, PADL 2011

T. Petricek, D. Syme. *The F# Computation Expression Zoo*, PADL 2014

Type providers

Type providers

Data access

Source-specific vs. general-purpose providers

```
type WorldBank = WorldBankProvider<Asynchronous = true>
let data = WorldBank.GetDataContext()
let countries = [ data.Countries.Sweden; ... ]
```

Language and platform integration

```
type j = TypeScript.Api<"jquery.d.ts">
```

D. Syme et al. *Themes in Information-Rich Functional Programming for Internet-Scale Data Sources,* DDFP 2013

Not your grandma's type safety

ML type system has its merits here...

...just different than you thought!

Invaluable when **writing code**

Safety guarantees still exist

Well-typed programs **don't go wrong?**

Handling **data-source changes** (help?)

Importing **unsound types** (blame?)

Design considerations

Orthogonal design

- Async workflows, type providers, meta-programming
- Independent features, **play well together**

What can be **in the library**

- All minimal **syntactic** extensions
- Prefer library **without making code ugly**
- Type providers, computation expressions, quotations

Conclusions

We need more **case studies**!

one + one ≥ two

Type safety is relative

ML-style languages are nice!

Get in touch: tomas@tomasp.net | @tomaspetricek | @dsyme

Read my rant: http://tomasp.net/academic/papers/philosophy-pl