TREBALL DIRIGIT:

HASH CODE: 7472

ÍNDEX DEL TREBALL

ESTUDI DEL LLENGUATGE

1. INTRODUCCIÓ
2. HISTÒRIA DEL LLENGUATGE
3. ANÀLISI DEL LLENGUATGE
   - -PROPÒSITS DEL LLENGUATGE
   - -PARADIGMES DE PROGRAMACIÓ
   - -SISTEMA D'EXECUCIÓ
   - -APLICACIONS DEL LLENGUATGE
   - -SISTEMES DE TIPUS
   - -CARACTERÍSTIQUES DEL LLENGUATGE
   - -PONY I ALTRES LLENGUATGES
4. EXEMPLES DE CODI
5. CONCLUSIONS PERSONALS

ESTUDI BIBLIOGRÀFIC

1. ANÀLISI DE LES FONTS
2. REFERÈNCIES I CITACIONS

Actors in Pony

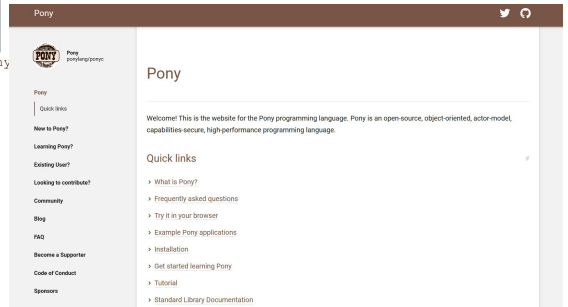
```
actor Act
    let env : Env
    let name : String

actor Main
                                  :nv, s: String) =>
  new create(env: Env) =>
    let act1 : Act = Act(env,"   A")
    let act2 : Act = Act(env,"   B")
    let act3 : Act = Act(env,"   C")  :t(name)

    act1.poke()
    act2.poke()
    act3.poke()
```

Code in `1_Actors/ABC.pony`

14



INTRODUCCIÓ I HISTÒRIA DEL LLENGUATGE

CARACTERÍSTIQUES:
- CODI OBERT
- SEGUR
- ESCALABE
- CONCURRENT
- MULTIPARADIGMÀTIC
- D'ALT NIVELL
- TOLERANT A LES FALLADES

Història:

Imperial College London and Sophia Drossopoulou

creació → 06/2014
compilació → 11/2014
publicació → 05/2015

nom amb dedicatòria

ANÀLISI DEL LLENGUATGE (I): PROPÒSITS I PARADIGMES

PROPÒSITS

- SEGURETAT
  - Memòria
  - Concurrència
- CONCURRÈNCIA
- PRODUCTIVITAT I RENDIMENT
- INTEROPERABILITAT
- LLEGIBILITAT

PARADIGMES PROGRAMACIÓ

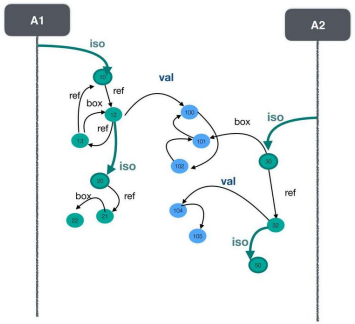
- ORIENTADA A OBJECTES
- FUNCIONAL
- CONCURRENT
- DE TIPUS ESTÀTIC

ANÀLISI DEL LLENGUATGE(II): SISTEMES D'EXECUCIÓ

- REFERENCE CAPABILITIES (CAPACITATS DE REFERÈNCIA)

reference capabilities
(adapted from morning paper)
- and omitting **trn** -

	holder may Read, Write?	local alias might	global alias might	holder may send?
iso	RD, WR	–	–	✅
ref	RD, WR	RD, WR	–	
val	RD	RD	RD	✅
box	RD	RD, WR	RD	
tag	–	RD, WR	RD, WR	✅



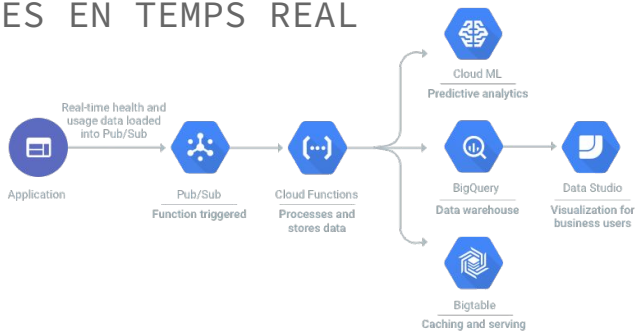
- RECOLLIDA DE ESCOMBRARIES

- COMPILADOR

ANÀLISI DEL LLENGUATGE (III): APLICACIONS

SISTEMES DISTRIBUÏTS

SISTEMES DE PROCESSAMENT DE DADES EN TEMPS REAL



APLICACIONS I SERVEIS WEB

JOCS I SIMULACIÓ

ANÀLISI DEL LLENGUATGE (IV): SISTEMES DE TIPUS

1-PRIMITIVES

2-CLASSES

3-STRUCTS

4-ACTORS

5-ÀLIES

6-EXPRESSIONS

7-INFERÈNCIA TIPUS

ANÀLISI DEL LLENGUATGE(V): ALTRES CARACTERÍSTIQUES

DETERMINISME

DADES AÏLLADES SEGURES

RENDIMENT, SEGURETAT I CONCURRÈNCIA

UN FIL PER ACTOR

DADES INMUTABLES SEGURES

FALTA D'ESTABILITAT DE L'API

MANCA BIBLIOTEQUES

CORBA APRENENTATGE DURA

PETITA COMUNITAT D'USUARIS

ANÀLISI DEL LLENGUATGE (VI): PONY VS ERLANG

1. MODELS DE CONCURRÈNCIA
2. TIPOLOGÍA (estàtica vs dinàmica)
3. SEGURETAT
4. RENDIMENT (eficiència vs tolerància)
5. SUPORT, EINES I LLIBRERIES

EXEMPLES DE CODI(I)

ponyc / examples / helloworld / main.pony

```
1   actor Main
2     new create(env: Env) =>
3       env.out.print("Hello, world.")
```

ponyc / examples / circle / main.pony

```
3   class Circle
4     var _radius: F32
5
6     new create(radius': F32) =>
7       _radius = radius'
8
9     fun ref get_radius(): F32 =>
10      _radius
11
12    fun ref get_area(): F32 =>
13      F32.pi() * _radius.pow(2)
14
15    fun ref get_circumference(): F32 =>
16      2 * _radius * F32.pi()
```

```
18  actor Main
19    new create(env: Env) =>
20
21      for i in Range[F32](1.0, 101.0) do
22        let c = Circle(i)
23
24        var str =
25          recover val
26            "Radius: " + c.get_radius().string() + "\n" +
27            "Circumference: " + c.get_circumference().string() + "\n"
28            "Area: " + c.get_area().string() + "\n"
29          end
30
31        env.out.print(str)
32      end
```

Podem observar com és la creació dels diferents elements i les paraules clau (en color vermell) necessàries per fer-ho

EXEMPLES DE CODI(II)

ponyc / examples / mixed / main.pony

```
3   actor Worker
4     var _env: Env
5
6     new create(env: Env) =>
7       _env = env
8
9       var a: U64 = 86028157
10      var b: U64 = 329545133
11
12      var result = factorize(a*b)
13
14      var correct =
15        try
16          (result.size() == 2) and
17            (result(0)? == 86028157) and
18            (result(1)? == 329545133)
19        else
20          false
21        end
```

Veiem com els actors poden tindre variables, mètodes i com es privatitzen aquests camps

```
23    fun ref factorize(bigint: U64): Array[U64] =>
24      var factors = Array[U64](2)
25
26      if bigint <= 3 then
27        factors.push(bigint)
28      else
29        var d: U64 = 2
30        var i: U64 = 0
31        var n = bigint
32
33        while d < n do
34          if (n % d) == 0 then
35            i = i + 1
36            factors.push(d)
37            n = n / d
38          else
39            d = if d == 2 then 3 else (d + 2) end
40          end
41        end
42
43        factors.push(d)
44      end
45
46      factors
```

CONCLUSIONS I CRÍTICA PERSONAL

LA FI JUSTIFICA ELS MITJANS?

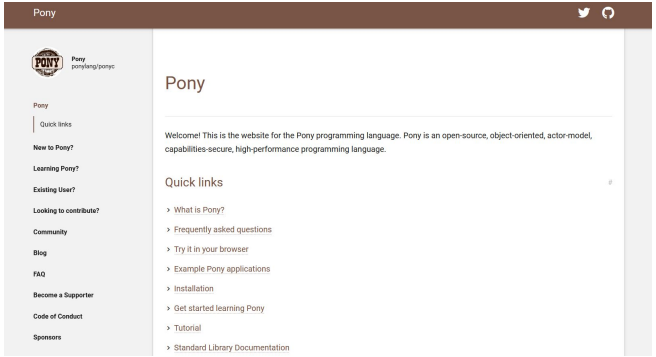
QUE N'OPINEN ELS USUARIS?

MEREIXEN UNA OPORTUNITAT?

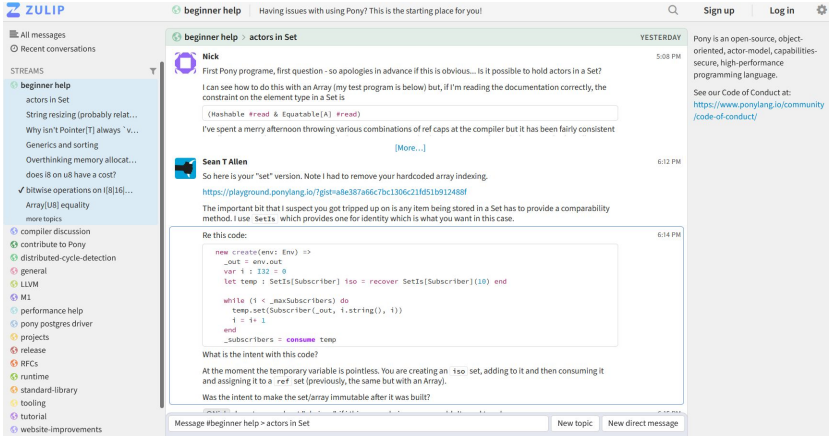
ÉS JUST CRITICAR PER LA POPULARITAT ACTUAL?

ESTUDI BIBLIOGRÀFIC

PÀGINA OFICIAL



CRÍTIQUES DE LA COMUNITAT



ENTREVISTES

Summary

Sophia Drossopoulou gives an overview of Pony's programming model, actors, and causality. She introduces the type system, how it is used to allow actors to send mutable state while also avoiding data races, and how the type system is used so as to allow the actors to perform garbage collection fully concurrently with one another and with normal execution.

- Dificultat en la recerca contrastada
- Ús d'eines recents d'extracció informativa

GRÀCIES PER LA VOSTRA ATENCIÓ!