A type inference algorithm for OO dynamic languages supporting blocks and generic types

Javier Fernandes[1,2] Nicolás Passerini[1,2,4]
Pablo Tesone[3,1,2,4]

[1]Universidad Nacional de Quilmes

[2]Universidad Nacional de San Martin

[3]Universidad Nacional del Oeste

[4]Universidad Tecnológica Nacional - F.R. Buenos Aires.

Workshop de Ingeniería en Sistemas y Tecnologías de la Información
28/11/2014

Agenda

¿Por qué es difícil aprender OOP?

- Enfoque en un lenguaje particular
- Demasiados conceptos

```
package examples;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

- Entornos de desarrollo pobres

¿Por qué es difícil aprender OOP?

- Enfoque en un lenguaje particular
- Demasiados conceptos

```
package examples;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

- Entornos de desarrollo pobres

Conclusión: deficiencias de aprendizaje

- Bajos niveles de aprobación
- Se propician malas prácticas
- Poca comprensión de los fundamentos del paradigma

Nuestra primera propuesta

1. Pensar en el recorrido[1]
   - Introducir los conceptos gradualmente.
   - Arrancar por los fundamentales:
     objeto - mensaje - referencia - polimorfismo
   - Postergar los accesorios:
     clases - herencia - ...
2. Construir una herramienta específica
   - Lenguaje de programación
   - Entorno de desarrollo

[1] Paper de Lombardi, Passerini y Cesario, 2007

Nuestra primera propuesta

1. Pensar en el recorrido[1]
   - Introducir los conceptos gradualmente.
   - Arrancar por los fundamentales:
     objeto - mensaje - referencia - polimorfismo
   - Postergar los accesorios:
     clases - herencia - ...
2. Construir una herramienta específica
   - Lenguaje de programación
   - Entorno de desarrollo

[1] Paper de Lombardi, Passerini y Cesario, 2007

Ozono

- Entorno de objetos sin clases
- Basado en Pharo
  - Basado en imagen (no archivos)
  - Dinámico
  - Diagrama automático de objetos

7 años de aprendizaje

Ozono fue una linda experiencia:

- Subieron los niveles de aprobación (de 40 - 50% a 80 - 90%)
- Exportado a otras universidades: UNQ, UNSAM, UNO, FRD, ...
- Comunidad $>$ 30 desarrolladores
- Proyectos de investigación

Peeero...

7 años de aprendizaje

- No provee una transición a clases
- Falencias en el ambiente
  - Muy atado a pharo (ej: Debugger)
  - Herramientas no preparadas para el aprendizaje
- A veces... extrañamos los tipos
  - Las herramientas no pueden ayudar al programador
  - Errores simples son difíciles de detectar.
- Lejano al *mainstream*
  - Basado en imagen
  - No usa herramientas estándares de la industria

7 años de aprendizaje

- No provee una transición a clases
- Falencias en el ambiente
  - Muy atado a pharo (ej: Debugger)
  - Herramientas no preparadas para el aprendizaje
- A veces... extrañamos los tipos
  - Las herramientas no pueden ayudar al programador
  - Errores simples son difíciles de detectar.
- Lejano al *mainstream*
  - Basado en imagen
  - No usa herramientas estándares de la industria

7 años de aprendizaje

- No provee una transición a clases
- Falencias en el ambiente
  - Muy atado a pharo (ej: Debugger)
  - Herramientas no preparadas para el aprendizaje
- A veces... extrañamos los tipos
  - Las herramientas no pueden ayudar al programador
  - Errores simples son difíciles de detectar.
- Lejano al *mainstream*
  - Basado en imagen
  - No usa herramientas estándares de la industria

7 años de aprendizaje

- No provee una transición a clases
- Falencias en el ambiente
  - Muy atado a pharo (ej: Debugger)
  - Herramientas no preparadas para el aprendizaje
- A veces... extrañamos los tipos
  - Las herramientas no pueden ayudar al programador
  - Errores simples son difíciles de detectar.
- Lejano al *mainstream*
  - Basado en imagen
  - No usa herramientas estándares de la industria

Wollok the language

- Integra clases y objetos
- Sintaxis educativa
  - Simple
  - Énfasis en los conceptos a transmitir (ej: method, val/var).
  - Concisa (ej: literales).
- Inferencia de tipos
- Basado en archivos

¿Qué debería tener un entorno de desarrollo?

- *Autocomplete*
- Navegación (ej: "Ir a la implementación")
- Búsqueda inteligente (ej: referencias, implementaciones, usos)
- Wizards
- Refactors automáticos
- Herramientas de debugging
- Herramientas de trabajo en grupo
- Herramientas de visualización de código

Conclusions

- The algorithm provides useful type information.
- Computes generic types for blocks and collections.
- Supports recursive programs.
- Can type parameters, instance variables, local variables and any expression.
- Works on Pharo programs without modifications.
- Useful testbench for testing new ideas

Further Work

- Metrics (both about precision and performance).
- Integration to Pharo tools.
- Scalability and performance in order to analyse bigger programs.
- Formal proofs
- Use it to analyse Pharo core libraries.