Investigacion de C++

Edinson Sanchez//Ruso Perez

19 de octubre de 2013

1. Introduccion

Your text goes here.

1.1. subtema

More text.

2. caracteristicas del lenguaje

revisar el libro de git

3. Historia del lenguaje

3.1. Carl Hewitt y el proyecto Planner

En 1971, Gerald Jay Sussman, Drew McDermott y Eugene Charniak desarrollaron un sistema llamado Micro-Planner, una parcial e insatisfactoria implementación de Planner. Sussman y Carl Hewitt trabajaron, junto con otros, en Muddle (luego MDL), una extensión de Lisp que formo un componente del proyecto Planner de Hewitt. En 1972, Sussman y McDermott desarrollaron un lenguaje basado en Lisp llamado Conniver.

En noviembre de 1972, Hewitt y sus estudiantes en el MIT desarrollaron el modelo Actor en computación, el cual fue propuesto como una solución a los problemas con Planner. Steele y Sussman luego decidieron implementar una versión del modelo Actor en MacLisp. Comenzaron a desarrollar mecanismos para crear actores y enviar mensajes.

3.2. Implementacion del modelo Actor y el nacimiento de Scheme

Sussman y Guy L. Steele luego implementaron el modelo Actor en el cálculo lambda, llamando a su sistema de modelación: Schemer. Este nombre fue luego reducido a 'Scheme' ya que hicieron uso del sistema operativo ITS, que limitaba los nombres de fichero a seis caracteres. Luego, ambos concluyeron que los Actores y los Closures eran, para propósitos de sus investigaciones, un concepto idéntico y procedieron a eliminar

el código redundante. Luego de la eliminación, se dieron cuenta que habían creado un pequeño, y muy capaz, dialecto de Lisp.

Hewitt estuvo escéptico de las capacidades del cálculo lambda como base de la programación y dijo, "La actual situación es que el cálculo lambda es capaz de expresar ciertos tipos de estructuras de control paralelo y secuencial, pero, en general, no es capaz de expresar la concurrencia que expresa el modelo Actor. Por otro lado, el modelo Actor es capaz de expresar todo lo del cálculo lambda, y mas."Hewitt también ha sido fuerte en las criticas sobre Scheme en base a su falta de manejo de excepciones y su dependencia a lo que se llaman continuation functions.

3.3. Publicación y Estandarización

Scheme es un lenguaje de programación que enfatiza la aplicación de funciones, o lenguaje funcional, que tiene sus raíces en el lenguaje de programación Lisp, siendo un dialecto de este. Scheme tuvo sus inicios en el periodo de 1975 hasta 1980, cuando Guy L. Steele y Gerald Jay Sussman, del MIT AI Lab, publicaron nueve artículos llamados Lambda Papers durante este tiempo.

Aunque una de las características mas importantes de destacar sobre Scheme es el minimalismo, Sussman y Steele expresaron que esta no fue una meta intencional durante su creación. "Nosotros en realidad estábamos tratando de construir algo complicado y rebuscado, favorablemente, habíamos creado algo que, además de cumplir con todas nuestras metas, era mucho más simple de lo intencionado... Nos dimos cuenta que el cálculo lambda puede servir como la base de un poderoso lenguaje de programación."(Gerald Jay Sussman y Guy L. Steele, Jr. (Diciembre 1998). "The First Report on Scheme Revisited").

Scheme tiene un estándar oficial de la IEEE y un estándar de facto llamado Revised n-th Report on the Algorithmic Language Scheme (comunmente abreviado RnRS). El estándar mas implementado hasta la fecha es el R5RS de 1998, y el nuevo estándar, R6RS, fue ratificado el 2007.

4. Tutorial de Instalacion

revisar el wikibook de latex hacer etiquetas para mencionar

5. Hola Mundo y otros Programas Introductorios

revisar el sourcecode listing

```
begin
{ do nothing }
end;
Write('Case insensitive');
Write('Pascal keywords.');
```