Unidad 1 - Compiladores

Marco Flores Nuñez

18 de marzo de 2013

1. Historia de los lenguajes de programación

La historia de los lenguajes de programación abarca un gran período de tiempo y un gran número de avances. Ya que un lenguaje de programación representa una forma de transcribir instrucciones para que éstas sean comprendidas por una máquina, podemos remontar su historia hasta mucho antes de la aparición de la primera computadora moderna.

1.1. Antes de 1940

Desde mucho antes que se desarrolle la primera copmutadora moderna las personas han diseñado formas de automatizar procesos o codificar información. Como ejemplo tenemos al telar de Jacquard, que utilizaba agujeros en tarjetas para representar movimientos de un brazo de tejido y así generar patrones de tejido automáticamente.

En las primeras décadas del siglo XX los cálculos numéricos se hacían utilizando el sistema decimal. Eventualmente se descubrió que la lógica no sólo podía ser representada usando palabras, si no también números. La máquina de Turing surgió como una abstraccion de la forma en que operaban las máquinas de marcado de cinta, en ese entonces usadas por compañías telefónicas. La máquina de Turing sirvió como base para la idea de almacenamiento de programas como información en la aruitectura de computadores von Neumann, al representar a una máquina con un número finito. Sin embargo la máquina de Turing no sirve como base para el diseño de lenguajes más altos, sino como un ejercicio académico para el análisis de algoritmos.

Desde el comienzo, los lenguajes de programación eran muy dependientes del hardware. FORTRAN incluyó un par de palabras en inglés (IF, GOTO de "go toz CONTINUE).

1.2. La década de 1940

Surgieron las primeras computadores modernas. Su limitada velocidad y capacidad de memoria forzaban a los desarrolladores a crear programas utilizando un lenguaje de ensamblador específico a cada máquina.

1.3. Los '50s y '60s

Surgieron los tres primeros lenguajes de programación modernos que aún son usados hoy:

- FORTRAN(1955) o "FORmula TRANslator"
- LISP(1958) o "LISt Processor"
- COBOL(1959) o ÇOmmon Business Oriented Language"

Otro hito a fines de los 50 fue la publicación del lenguaje ALGOL (ALGOrithmic Language); este lenguaje consolidaba varias ideas populares en ese entonces, incluyendo:

- estructura de bloques anidados
- lexical scoping; es decir la independencia e invisibilidad entre las variables dentro de distintos bloques

También fue una innovación la forma en que describió el lenguage; usando la notación Backus-Naur Form (BNF). A partir de entonces casi todos los lenguajes han sido descritos usando esta notación.

1.4. 1968-1979

En este período se crearon la mayoría de paradigmas usados hoy en día:

- Simula: creado a fines de los 60, fue el primer lenguaje diseñado para soportar programación orientada a objetos.
- C: creado por Dennis Ritchie y Ken Thompson entre 1969 y 1973.
- Smalltalk: diseñado desde cero para ser un lenguaje orientado a objetos.
- Prolog: el primer lenguaje de programación lógica.

Cada uno de estos lenguajes desencadenó en una gran familia de descendientes.

En esta época también surgió un debate considerable a favor de la "programación estructurada", que favorecía el uso de bloques y/o indentación en vez de la sentencia "goto". A pesar de que esta controversia fue de gran escala en su tiempo, hoy en día casi todos los programadores están de acuerdo en que la programación estructurada es de mayor ventaja y el uso de "goto.[es] considerado una mala práctica (en aquellos lenguajes que aún lo soportan).

Algunos lenguajes importantes de este período incluyen a:

- Logo
- C
- Pascal
- Smalltalk
- Prolog
- Scheme
- SQL, que inicialmente fue un lenguaje de sólo consulta, más adelante se expandió con funcionalidades de programación.

1.5. La década de 1980

En esta década hubo una gran consolidación de los lenguajes imperativos. En vez de crear nuevos paradigmas, se fortalecieron las ideas de la década anterior. C++ le añadió programación orientada a objetos a C.

Una tendencia importante en el diseño de lenguajes fue el enfoque sobre la programación de sistemas a gran escala a través del uso de módulos, o unidades de código organizacionales a gran escala. Modula, Ada y ML desarrollaron sistemas de módulos notables en los 80s.

Esta década también trajo consigo avances en la implementación de los lenguajes de programación. Se propuso que el hardware debía ser diseñado para los compiladores, en vez de para los programadores de lenguaje ensamblador. Este movimiento generó un gran interés en tecnologías de compilación para lenguajes de mayor orden.

Algunos lenguajes importantes de este período incluyen a:

- C++ (primero conocido como C with classes)
- Ada
- Common Lisp
- MATLAB
- Objective-C
- Erlang
- Perl

1.6. Los 90s

El rápido crecimiento de la Internet en esta década fue el siguiente hito en la historia de los lenguajes de programación. La Internet creó una oportunidad para que nuevos lenguajes sean adoptados al abrir una plataforma totalmente nueva. Particularmente, Java se abrió paso debido a su fácil integración con el navegador Netscape, y varios lenguajes de scripting se hicieron populares para el desarrollo de aplicaciones para servidores web. En esta década se vio el crecimiento de los lenguajes funcionales.

Una de las filosofías principales se enfocaba en la productividad del programador. Surgieron muchos lenguajes de rapid application development.º desarrollo rápido de aplicaciones (RAD) que usualmente incluían un IDE, recolección de basura y descendían de otros lenguajes; como ejemplo tenemos a Object Pascal, Visual Basic y Java. Todos ellos eran orientados a objetos.

Más radicales e innovadores que los lenguajes RAD eran los lenguajes de scripting. Estos no descendían directamente de ningún otro lenguaje y poseían nuevas sintaxis y características. Muchos consideran que estos son incluso más rápidos que los lenguajes RAD, pero usualmente debido a opciones de diseño que hacen que programas pequeños sean simples de escribir y mantener mientras que programas grandes son lo opuesto. Aún así, los lenguajes de scripting llegaron a convertirse en los más utilizados con respecto a la web.

Algunos lenguajes importantes de este período incluyen a:

- Haskell
- Python
- Visual Basic
- HTML (lenguaje de marcado)
- Ruby
- Java
- JavaScript
- PHP

2. Breve revisión de los paradigmas de programación

Un paradigma de programación es un estilo fundamental de programación. Se basan en distintos modelos de computación: máquina de Turing para programación orientada a objetos e impoerativa, cálculo lambda para programación funcional y lógica de primer orden para programación lógica.

2.1. Lenguajes procedurales

El paradigma procedural está basado en llamadas a procedimientos. Los procedimientos también puede ser conocidos como rutinas, subrrutinas, métodos o funciones. Estos procedimientos simplemente contienen un conjunto de instrucciones que deben ser llevadas a cabo.

Durante la ejecución de un programa cualquier procedimiento puede ser llamado arbitrariamente, inclusive por otros procedimientos o por sí mismo.

Entre estos lenguajes se encuentran C++, C, Fortran y Pascal.

2.2. Lenguajes orientados a objetos

Los lenguajes orientados a objetos siguen un paradigma que busca representar conceptos como .ºbjetosçon campos de información (o atributos que describen al objeto) y procedimientos asociados llamados "métodos".

Un programa orientado a objetos puede ser visto como una colección de objetos que interactúan entre sí, a diferencia del modelo convencional, en el que un programa es un conjunto de rutinas a ejecutar. Cada uno de estos objetos es capaz de procesar información, ejectuar rutinas propias (métodos) y enviar mensajes a otros objetos.

Un concepto importante es aquel de çlase". Una clase es el arquetipo a partir del cual se pueden crear objetos, que son a su vez instancias de una clase. La clase determina los atributos y métodos que puede tener un objeto determinado y define rutinas de construcción específicas.

2.3. Lenguajes funcionales

Los lenguajes funcionales siguen el paradigma en el cual la computación se basa en la evaluación de funciones matemáticas y evita el cambio de estado y variables.

El paradigma funcional está basado en el cálculo lambda, un sistema formal desarrollado en los años 30 para investigar la computabilidad de un problema y la definición, aplicación y recursividad de una función. Se puede ver a muchos lenguajes funcionales como extensiones del cálculo lambda.

En la práctica, la diferencia entre una función matemática y una función utilizada en un paradigma imperativo es que la segunda puede tener efectos secundarios que afecten al estado del programa. De este modo, una función imperativa puede retornar diferentes valores en distintas llamadas durante el programa. Una rutina funcional, en cambio, sólo depende de las entradas que recibe y, para el mismo conjunto de entradas siempre retornará la misma salida.

2.4. Lenguajes declarativos y no algorítimicos

Los lenguajes declarativos buscan describir *qué* computación debe ser realizada y no *cómo* debe realizarse. El *cómo* se le deja a la implementación del lenguaje.

Recientemente se ha volcado interés en este paradigma ya que podría simplificar considerablemente la creacion de programas paralelos.

2.5. Lenguajes de scripts

Los lenguajes de scripts son aquellos que automatizan la ejecución de tareas que, de otra forma, requerirían ser ejecutadas manualmente una por una por un operador humano.

Entre los entornos que pueden ser automatizados por un script están las páginas web, los navegadores web, la shell de un sistema operativo, etc.

Un lenguaje de scripting se caracteriza por las siguientes propiedades:

- Facilidad de uso.
- Interpretables, no necesitan ser compilados, lo cual reduce el tiempo entre el scripting y la ejecución.
- Estructura relativamente flexible.

3. Noam Chomsky

Es un lingüista estadounidense, actualmente profesor del Departamento de Lingüística y Filosofía en MIT, en donde ha trabajo por más de 50 años.

A pesar de ser lingüista de profesión, ha influido en el campo de la Ciencia de la Computación al describir la jerarquía Chomsky de clases de gramáticas formales. Esta jerarquía es enseñada ya que ofrece iluminación sobre los diferentes tipos de lenguajes formales.