Notación Backus-Naur

Backus-Naur Form

Autor 1: Julián Esteban Collazos Toro
Ingeniería en sistemas y computación, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: j.collazos@utp.edu.co

***Resumen*— Trata de un metalenguaje, que plantea normas para redacción y sintaxis, usadas dentro de los lenguajes de programación, en cuanto a nosotros respecta. "El objetivo fue que esa gramática les permitiera crear una base de instrucción para describir y modelar distintos tipos de textos, a saber: documentos que estén bajo formato, conjunto de instrucciones de programación, protocolos de comunicación, lenguajes de programación, notación para las gramáticas y sintaxis de los lenguajes de programación de la computadora, etc." [1] En otras palabras, define la manera en que debemos escribir dentro de un lenguaje, de forma general.**

***Palabras clave*—Metalenguaje, Sintaxis, Programación, Gramática**

***Abstract* It is a metalanguage, which raises standards for writing and syntax, used within programming languages, as far as we are concerned. "The objective was that this grammar would allow them to create an instruction base to describe and model different types of texts, namely: documents that are in format, set of programming instructions, communication protocols, programming languages, notation for grammars and syntax of computer programming languages, etc. "[1] In other words, it defines the way in which we should write within a language, in general.**

***Key Word* — Metalanguage Syntax, Programming, Grammar**

I. INTRODUCCIÓN

Hay reglas para casi todo lo que conocemos y hacemos, lo mismo sucede con la escritura, pero no cualquier escritura, si no, la escritura en programación. Usando especificaciones, símbolos, expresiones, entre otras; que definen lo que se debe hacer, como escribir, y que es lo que contiene dicha expresión.
Esta notación también usa la jerarquía de Chomsky, jerarquía que explicare un poco dentro del contenido. Sin más preámbulo, comencemos.

Inducción Matemática, mismo que sirve para probar o establecer que una determinada propiedad se cumple para todo número natural

II. CONTENIDO

1. Historia

La idea de transcribir la estructura del lenguaje con reglas de reescritura se remontan cuando menos al trabajo del gramático indio Panini (hacia el 460 a. C.), que la utilizó en su descripción de la estructura de palabras del idioma sánscrito[…] Lingüistas estadounidenses como Leonard Bloomfield y Zellig Harris llevaron esta idea un paso más adelante al tratar de formalizar el lenguaje y su estudio en términos de definiciones formales y procedimientos (1920-1960). Noam Chomsky, maestro de lingüística de alumnos de teoría de la información del MIT, combinó la lingüística y las matemáticas, tomando esencialmente el formalismo de Axel Thue como la base de su descripción de la sintaxis del lenguaje natural. También introdujo una clara distinción entre reglas generativas (de la gramática libre de contexto) y reglas transformativas (1956). John Backus, un diseñador de lenguajes de programación de IBM, adoptó las reglas generativas de Chomsky para describir la sintaxis del nuevo lenguaje de programación IAL, conocido en la actualidad como ALGOL 58 (1959), presentando en el primer Congreso de Computación Mundial (World Computer Congress) el artículo «The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference».

Peter Naur, en su reporte sobre ALGOL 60 de 1963, identificó la notación de Backus como la Forma Normal de Backus (Backus Normal Form), y la simplificó para usar un conjunto de símbolos menor,

pero a sugerencia de Donald Knuth, su apellido fue agregado en reconocimiento a su contribución, reemplazando la palabra «Normal» por Naur, dado que no se trata de una forma normal en ningún sentido, a diferencia, por ejemplo de la Forma Normal de Chomsky

2. **Teoría**

Entenderemos lenguaje como conjunto de símbolos. En este caso serán llamados letras, que formarán palabras y estas, a su vez oraciones, y todo esto organizado por un diccionario. Este es el lenguaje natural, pues se forman oraciones no construidas, simplemente se dan, con reglas básicas.

Por el contrario, el lenguaje formal trata de propiedades o formulas, que definan las oraciones. Por lo que aquí una palabra tendrá el mismo significado siempre, de lo cual surgen las gramáticas libres de contexto.

Ahora bien, todos estos son conceptos usados dentro la teoría de la notación de Backus-Naur. Esta teoría está basada en la jerarquía de Chomsky la cual habla de distintas gramáticas: sin restricciones, sensitivas al contexto, libres de contexto y regulares. Backus-Naur solo trata la gramática libre de contexto, en los que se pueden ingresar proposiciones dentro de proposiciones sin tener en cuenta su contexto.

La notación de Backus-Naur tiene las siguientes clausulas:

Literal: trata de las palabras clave que forman el cuerpo de la sentencia. Aquí no se utilizan etiquetas.

Obligatoria: es la parte que debe especificarse para que la sentencia sea correcta. Debe escribirse dentro de los siguientes símbolos: <obligatoria>.

Opcional: es la parte que puede o no estar, y la secuencia no se verá afectada. Se encierra dentro de corchetes [Opcional]

Iterable: esta cláusula puede aparecer varias veces dentro de la sentencia. Se encierra dentro de llaves {Iterable}.

Exclusión: representa un conjunto de valores mutuamente excluyentes. En este caso solo se puede escoger uno de los valores. Se separan por una barra vertical. valor1|valor2.

3. **Ejemplos:**

dirección postal de los EE.UU.

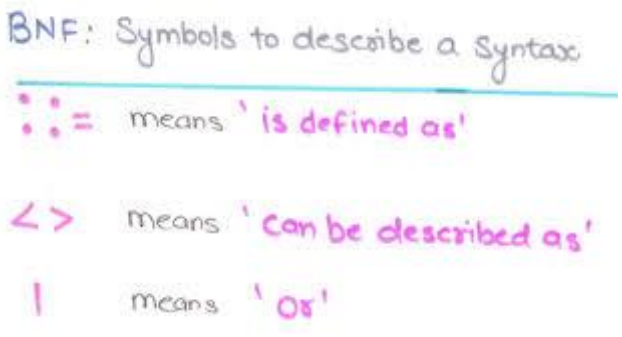
<dirección postal>:: = <**nombre**> <dirección> <**apartado postal**>

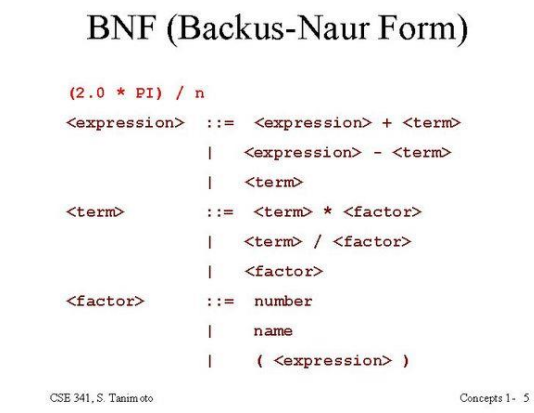
<**nombre**> ::= <**personal**> <**apellido**> [<**trato**>] <**EOL**>
| <**personal**> <**nombre**>

<personal> ::= <primer nombre> | <inicial> "."

<**direccion**> ::= [<**dpto**>] <**numero de la casa**> <**nombre de la calle**> <**EOL**>

<**apartado postal**> ::= <**ciudad**> "," <código estado> <código postal> <**EOL**>






```
<syntax> ::= <rule> [<syntax>]
<rule> ::= <whitespace> "<" <rule-name> ">" <whitespace> "::="
   <expression> <whitespace> <line-end>
<expression> ::= <whitespace> <or-expression>
<or-expression> ::= <whitespace> <list-expression> [ "|"
   <or-expression> ]
<list-expression> ::= <whitespace> ("<" <rule-name> ">"
   | <QUOTE> <text> <QUOTE> | "(" <expression> ")" | "[" <expression>
   "]") [<list-expression>]
<whitespace> ::= [" " <whitespace>]
<line-end> ::= [<whitespace>] <EOL> [<line-end>]
```

4. Biografías

4.1 John Backus

Antes de concebir y desarrollar el Fortran, los ordenadores funcionaban gracias a unas grandes

tarjetas perforadas a mano con las instrucciones que le permitía operar. Backus llevó el lenguaje de la programación a un nivel superior y más intuitivo, con comandas que permitían a la máquina generan sus propios códigos.

Backus, nacido en Wilmington (Delaware) en 1924, decía que su invención fue fruto "de ser vago". "No me gusta escribir programas", comentó ya en 1979, por eso decidió empezar a trabajar en un sistema que permitiera programar las computadoras con facilidad. Entonces utilizaba un IBM 701, que se usaba para computar las trayectorias de los misiles.

No se puede decir que su formación fuera brillante. Al contrario. Quizá por falta de motivación. Dejó la Universidad de Virginia a los seis meses de su ingreso y tras entrar en el ejército intentó con la medicina, pero se dio cuenta enseguida que su vocación estaba en la ingeniería. A partir de ese momento, se dedicó a estudiar matemáticas y obtuvo un *master* por la Universidad de Columbia. Al poco de graduarse, era contratado por la compañía IBM.

Backus entraba así en el mundo de las computadoras, trabajando entre el selecto grupo de ingenieros dedicado al Selective Secuence Electronic Calculator. En 1954, la compañía electrónica le encargó crear un grupo para diseñar un sistema más sencillo de programación, del que nació el Fortran, que es la abreviación en inglés de Formula Translation. Muchos se mostraron escépticos.

Sin embargo, con el paso de los años, su lenguaje demostró que funcionaba y del Fortran surgieron otros sistemas aún en uso. Como dicen los expertos del mundo informático, el Fortran era el primer lenguaje informático comprensible para los humanos, muy similar a fórmulas algebraicas que científicos e ingenieros utilizaban en la vida diaria.

La invención de Backus, que abrió en 1957 las puertas a la computación moderna, le vino acompañada años después de varios premios, como la Medalla Nacional de la Ciencia, en 1975, o el prestigioso Turing Award, que le concedió la Association for Computing Machinery en 1977. Y en 1993, dos años después de jubilarse de IBM, el programador fue galardonado con el Charles Stark Draper Prize, el máximo reconocimiento que concede la Academia Nacional de Ingeniería por el paso de gigante que Backus dio en el mundo de la informática.

4.2 Peter Naur

Pionero Danés de la informática. Contribuyó al desarrollo de la notación BNF (Backus Naur Form), utilizada en la descripción formal de la sintaxis de la mayoría de los lenguajes de programación, y a la creación del lenguaje de programación ALGOL 60.

Peter Naur nació el 25 de octubre de 1928 en Frederiksberg, cerca de Copenhague, Dinamarca.

Desde sus años de instituto mostró tener un gran interés por la astronomía, disciplina sobre la que comenzó sus estudios universitarios en 1947 y consiguió graduarse en 1949.

En los años 1950-51 realizó una estancia como estudiante investigador en el King's College de Cambridge, donde trabajó en la realización de un programa para el EDSAC que calculaba la perturbación de movimientos en los planetas.

Durante 1952-53 visitó varios observatorios astronómicos y laboratorios de desarrollo de computadores en los Estados Unidos y realizó una segunda estancia en Cambridge.

Entre los años 1953 y 1959 trabajó como asistente científico en el observatorio astronómico de Copenhague y colaboró como consultor en temas de lenguaje ensamblador y depuradores durante el desarrollo del primer computador danés, Dask, realizado en el laboratorio de computadores Regnecentralen.

Obtuvo en 1957 el título de Doctor en Astronomía.

En 1959 entra a formar parte de la plantilla del Regnecentralen, involucrándose en el desarrollo del lenguaje Algol 60. El Algol 60, representó un punto de referencia en el desarrollo de los lenguajes de programación, tanto por los conceptos que introducía como por el uso de una notación formal para su sintaxis, conocida como BNF y propuesta por Peter Naur y John Backus Hasta este momento podría decirse que Naur es un astrónomo con interés en computadores, pero a partir de aquí centra sus intereses solamente en los computadores.

Entre los años 1960 a 1967 se dedica al desarrollo de compiladores de Algol 60 y Cobol, introduciendo los conceptos de paginación, traducción multipaso y pseudo evaluación de expresiones.

En 1963 fue distinguido con la medalla G. A. Hagemann y en 1966 obtiene el premio Jens Rosenkjaer.

Naur introdujo el término datología para denominar a las ciencias de la computación. Ocupó la primera cátedra de datología de la Universidad de Copenhague en 1969 y permaneció en ella hasta 1998. Durante su carrera académica, con su trabajo contribuyó a la aportación de ideas y conceptos en el campo de la corrección de programas y en el diseño, producción y

mantenimiento de grandes sistemas software, siendo pionero en el área de ingeniería de software.
En 1986 recibió el premio de Pionero de los Computadores, concedido por la Computer Society del IEEE.

III. CONCLUSIONES

Es necesario determinar reglas para así hacer que todos, o al menos la mayoría, entendamos sobre que podemos trabajar, muchas veces estas reglas son una ayuda, como lo es este caso, la notación Backus-Naur (de ahora en adelante BNF), proporciona al programador una sintaxis fácil de leer, para así encontrar errores más rápido, entender qué es lo que se está haciendo, incluso para no perderse en las líneas de código.
Le debemos a Backus y a Naur, la sintaxis legible para programar.

REFERENCIAS

[1] https://es.wikipedia.org/wiki/Notaci%C3%B3n_de_Backus-Naur
http://lorien.die.upm.es/juancho/pfcs/DPF/capitulo2.pdf
https://elpais.com/diario/2007/03/21/agenda/1174431604_850215.html
http://www.mhi.fi.upm.es/spanish/htm/biografias-naur.htm
https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwj_-LTX-OfkAhVMdt8KHTKNBU4Qjhx6BAgBEAI&url=https%3A%2F%2Fulysses81.wordpress.com%2F2011%2F11%2F20%2Facceso-al-codigo-fuente-capitulo-3-representacion-de-los-elementos-del-lenguaje-y-la-gramatica%2F&psig=AOvVaw1-u4XIaPGJH-2THDyWKdvS&ust=1569362396804853
https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwid6qDo-OfkAhUNneAKHY5zA98Qjhx6BAgBEAI&url=http%3A%2F%2Fwww.tutorialsspace.com%2FProgramming-Languages%2F26-Formal-Method-Of-Describing-Syntax-Part-2-CFG-BNF.aspx&psig=AOvVaw1-u4XIaPGJH-2THDyWKdvS&ust=1569362396804853