Backus Naur (Backus Normal Form)

Backus Naur (Backus Normal Form)

Autor: Martin Alejandro Carvajal Rada

*Ingeniería de Sistemas, Universidad Tecnológica de Pereira, Pereira, Colombia*

Correo-e: m.carvajal1@utp.edu.co

***Resumen*— En este PAPER pueden evidenciar la teoría de Bakus Naur (Backus-Naur form), la biografía de Backus Naur, algunos ejemplos de su teoría y los impactos que esta hizo. El BNF se utiliza extensamente como notación para las gramáticas de los lenguajes de programación, de los sistemas de comando y de los protocolos de comunicación, así como una notación para representar partes de las gramáticas de la lengua natural (por ejemplo, el metro en la poesía de Venpa).**

***Palabras clave—* Backus Naur, Teoría, Notación, programación, biografia, BNF, PAPER, ABNF, Arpanet, Internet**

***Abstract*— In this paper you can prove the theory of Bakus Naur (Backus-Naur form), the biography of Backus Naur, some examples of his theory and the impacts it made. The BNF is widely used as a notation for the grammars of programming languages, command systems and communication protocols, as well as a notation to represent parts of the grammars of natural language (for example, the subway in Venpa’s poetry).**

***Key Word* — Backus Naur, Teoría, Notación, programación, biografia, BNF, PAPER, ABNF, Arpanet, Internet**

1. INTRODUCCIÓN

Las especificaciones técnicas de Internet a menudo necesitan definir una sintaxis formal y son libres de emplear cualquier notación que sus autores consideren útil. A lo largo de los años, una versión modificada de Backus-Naur Form (BNF), llamada BNF aumentada (ABNF), ha sido popular entre muchas especificaciones de Internet.

Equilibra la compacidad y la simplicidad, con un poder de representación razonable. En los primeros días del Arpanet, cada especificación contenía su propia definición de ABNF. Esto incluía las especificaciones de correo electrónico, [RFC733] y luego [RFC822], que llegaron a ser las citas comunes para definir ABNF. El documento actual separa esas definiciones para permitir la referencia selectiva.

Previsiblemente, también proporciona algunas modificaciones y mejoras que llegaron a ser las citas comunes para definir ABNF. El documento actual separa esas definiciones para permitir la referencia selectiva. Como era de esperar, también proporciona algunas modificaciones y mejoras. Las diferencias entre la BNF estándar y la ABNF implican reglas de nombres, repetición, alternativas, independencia de orden y rangos de valor. El Apéndice B proporciona definiciones de reglas y codificación para un analizador léxico central del tipo común a varias especificaciones de Internet. Se proporciona como una conveniencia y es por lo demás separada de la meta

1. CONTENIDO

Definicion de regla

1. Nombramiento de regla

El nombre de una regla es simplemente el nombre mismo; es decir, una secuencia de caracteres, comenzando con un carácter alfabético, y seguido por una combinación de alfabéticos, dígitos y guiones (guiones). NOTA: Los nombres de reglas son insensibles a mayúsculas y minúsculas Los nombres <rulename>, <Rulename>, <RULENAME>, y <rUlENamE> se refieren a la misma regla. A diferencia de la BNF original, no se requieren paréntesis angulares ("<", ">"). Sin embargo, se pueden usar paréntesis angulares alrededor de un nombre de regla siempre que su presencia facilite el discernimiento del uso de un nombre de regla.

Esto se limita típicamente a referencias de nombres de reglas en prosa de forma libre, o para distinguir reglas parciales que se combinan en una cadena no separada por espacio en blanco, como se muestra en la discusión sobre la repetición, abajo. Regla Forma Una regla se define por la siguiente secuencia: nombre = elementos crlf donde <nombre> es el nombre de la regla, <elementos> es uno o más nombres de reglas o especificaciones de terminal, y <crlf> es el indicador de fin de línea (retorno de carro seguido de alimentación de línea). El signo igual separa el nombre de la definición de la regla. Los elementos forman una secuencia de uno o más nombres de reglas y/o definiciones de valor, combinados según los diversos operadores definidos en este documento, como alternativa y repetición.

El nombre de una regla es simplemente el nombre en sí; es decir, una secuencia de caracteres, comenzando con un carácter alfabético, y seguido por una combinación de alfabéticos, dígitos, y guiones (guiones).

NOTA: Los nombres de regla son insensibles a mayúsculas y minúsculas

A diferencia de la BNF original, no se requieren paréntesis angulares ("<", ">"). Sin embargo, se pueden usar paréntesis angulares alrededor de un nombre de regla siempre que su presencia facilite el discernimiento del uso de un nombre de regla. Esto se limita típicamente a referencias de nombres de reglas en prosa de forma libre, o para distinguir reglas parciales que se combinan en una cadena no separada por espacio en blanco, como se muestra en la discusión sobre la repetición, abajo.

2. Forma de regla

Una regla se define por la siguiente secuencia:

name = elements crlf

donde <name> es el nombre de la regla, <elements> es una o más reglas nombres o especificaciones de terminales, y <crlf> es el final de línea indicador (retorno del carro seguido de alimentación de línea). El signo de igualdad separa el nombre de la definición de la regla. Los elementos formar una secuencia de uno o varios nombres de reglas y/o definiciones de valores, combinados según los distintos operadores definidos en el presente documento, como alternativa y la repetición.

Para facilitar la visualización, las definiciones de las reglas están alineadas a la izquierda. Cuando una regla requiere varias líneas, las líneas de continuación están sangradas. La alineación y sangría a la izquierda son relativas a las primeras líneas de las reglas de ABNF y no tienen por qué coincidir con el margen izquierdo del documento.

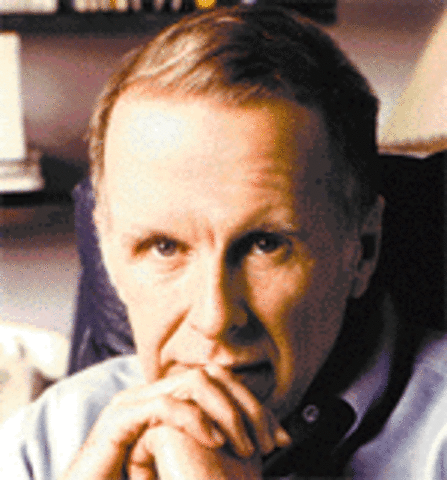
3. Codificaciones externas

Las representaciones externas de los caracteres de valor terminal variarán según las restricciones en el entorno de almacenamiento o transmisión. Por lo tanto, la misma gramática basada en ABNF puede tener múltiples codificaciones externas, como una para un entorno US-ASCII de 7 bits, otra para un entorno de octeto binario, y otra diferente cuando se utiliza Unicode de 16 bits. Los detalles de codificación están más allá del ámbito de ABNF, aunque el Apéndice A (Core) proporciona definiciones para un entorno US-ASCII de 7 bits como ha sido común a gran parte de Internet. Al separar la codificación externa de la sintaxis, se pretende que los entornos de codificación alternativos puedan utilizarse para la misma sintaxis.

4. Alternativas incrementales: Rule1 =/ Rule2

A veces es conveniente especificar una lista de alternativas en fragmentos. Es decir, una regla inicial puede coincidir con una o más alternativas, con definiciones de reglas posteriores que se añadan al conjunto de alternativas. Esto es particularmente útil para otras especificaciones independientes que derivan del mismo conjunto de reglas padre, como ocurre a menudo con las listas de parámetros.

Biografías:



**John Backus**

(Filadelfia, 3 de diciembre de 1924 - Oregón, 17 de marzo de 2007) fue un científico de la computación1​ estadounidense.

Ganador del Premio Turing en 1977 por sus trabajos en sistemas de programación de alto nivel, en especial por su trabajo con FORTRAN.

Para evitar las dificultades de programación de las calculadoras de su época, en 1954 Backus se encargó de la dirección de un proyecto de investigación en IBM para el proyecto y realización de un lenguaje de programación más cercano a la notación matemática normal. De ese proyecto surgió el lenguaje FORTRAN, el primero de los lenguajes de programación de alto nivel que tuvo un gran impacto, incluso comercial, en la emergente comunidad informática.

Tras la realización de FORTRAN, Backus fue un miembro muy activo del comité internacional que se encargó del proyecto de lenguaje ALGOL. En ese contexto propuso una notación para la representación de las gramáticas usadas en la definición de un lenguaje de programación (las llamadas gramáticas libres de contexto). Tal notación se conoce como Notación de Backus-Naur (Backus-Naur Form o BNF) y une al nombre de Backus al de Peter Naur, un informático europeo del comité ALGOL que contribuyó a su definición.

En los años 1970, Backus se interesó sobre todo por la Programación funcional, y proyectó el lenguaje de programación FP, descrito en el texto que le sirvió para ganar el premio Turing, ¿” Can Programming be Liberated from the Von Neumann Style?" Se trata de un lenguaje de uso fundamentalmente académico, que sin embargo animó un gran número de investigaciones. El proyecto FP, transformado en FL, se terminó cuando Backus se jubiló en IBM, en 1991.

John Backus falleció el sábado 17 de marzo de 2007, a la edad de 82 años en su casa en Ashland, Oregón por causas naturales, de acuerdo a la declaración de su familia.



**Peter Naur**

(Frederiksberg, 25 de octubre de 1928-3 de enero de 2016)1​ fue un científico danés pionero en informática y ganador del Premio Turing en 2005.

La N de la notación BNF, usada en la descripción de la sintaxis de la mayoría de los lenguajes de programación, se usa en alusión a su apellido. Naur contribuyó en la creación del lenguaje de programación Algol 60 en donde se introdujo por primera vez la noción de recursión.2​ Empezó su carrera como un astrónomo, pero su encuentro con las computadoras lo hizo cambiar de carrera. A Naur no le agradaba el término anglosajón Ciencias de la computación ("Computer Science" en inglés) y sugiere llamarlo Datalogía ("Datalogy" en inglés). Este término fue adoptado en Dinamarca y Suecia.

Trabajó en el Regnecentralen (empresa de computación danesa), en el Instituto Niels Bohr y en la Universidad Técnica de Dinamarca. De 1968 a 1998 trabajó como profesor en la Universidad de Copenhague. Es conocido por su crítica al uso de los métodos formales en programación. Así mismo, basado en su inclinación desde el empirismo, critica el uso que le dan los filósofos a la lógica para describir la ciencia. Critica igualmente a psicólogos que todavía se basan en teorías del conductismo y el constructivismo.

Se retiró en 1999 a la edad de 70 años.2​ En los últimos años estuvo desarrollando una teoría del pensamiento humano que denominó Teoría Sinapsis-Estado de Vida Mental ("A Synapse-State Theory of Mental Life" en inglés).

1. CONCLUSIONES

Primeramente, John Backus y posteriormente Peter Naur establecieron una sintaxis que (a nuestro juicio) responde a la lógica matemática (básicamente como todo lenguaje) y a una estructura algorítmica. 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. REFERENCIAS
2. <https://tools.ietf.org/html/rfc4234>
3. <https://www.elmundo.es/traductor/>
4. <https://es.wikipedia.org/wiki/John_Backus>
5. <https://es.wikipedia.org/wiki/Peter_Naur>
6. <https://www.fing.edu.uy/inco/cursos/teoleng/obligatorio/presentacion.pdf>
7. <http://www.garshol.priv.no/download/text/bnf.html#id2.2>.