Backus-Naur form

Fernando Estiven Castro Jaramillo – ingenieria en sistemas y computación

La forma de Backus-Naur (FBN), es una notación para especificar una gramática de tipo 2, llamada así en honor a John Backus, que la invento, y Peter Naur, quien la modifico para utilizarla en las especificaciones del lenguaje de programación ALGOL. La forma de Backus-Naur se emplea para especificar las reglas sintácticas de muchos lenguajes de programación, incluido el lenguaje Java.

**Historia:** La idea de transcribir la estructura del lenguaje con reglas de reescritura se remontan cuando menos al trabajo del gramático indio [Panini](https://es.wikipedia.org/wiki/Panini_(hindu)) (hacia el [460 a. C.](https://es.wikipedia.org/wiki/A%C3%B1os_460_a._C.)), que la utilizó en su descripción de la estructura de palabras del [idioma sánscrito](https://es.wikipedia.org/wiki/Idioma_s%C3%A1nscrito) (algunos incluso han sugerido renombrar BNF a Forma Panini-Backus). 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](https://es.wikipedia.org/wiki/Noam_Chomsky), maestro de lingüística de alumnos de teoría de la información del [MIT](https://es.wikipedia.org/wiki/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](https://es.wikipedia.org/wiki/John_Backus), un diseñador de lenguajes de programación de [IBM](https://es.wikipedia.org/wiki/IBM), adoptó las reglas generativas de Chomsky para describir la sintaxis del nuevo lenguaje de programación IAL, conocido en la actualidad como [ALGOL 58](https://es.wikipedia.org/w/index.php?title=ALGOL_58&action=edit&redlink=1) (1959), presentando en el primer Congreso de Computación Mundial ([World Computer Congress](https://es.wikipedia.org/w/index.php?title=World_Computer_Congress&action=edit&redlink=1)) 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](https://es.wikipedia.org/wiki/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.

Como ejemplo, considere este BNF para una [dirección postal](https://es.wikipedia.org/wiki/Direcci%C3%B3n_postal) de los [EE.UU.](https://es.wikipedia.org/wiki/Estados_Unidos)

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

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

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

<dirección> ::= [<**dpto**>] <número de la casa> <**nombre de la calle**> <**EOL**>

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

Esto se traduce a español como:

* Una dirección postal consiste en un nombre, seguido por una dirección, seguida por un apartado postal.
* Una parte «personal» consiste en un nombre o una inicial seguido(a) por un punto.
* Un nombre consiste de: una parte personal seguida por un apellido seguido opcionalmente por una jerarquía o el trato que se la da a la persona (Jr., Sr., o número dinástico) y un salto de línea (*end-of-line)*, o bien una parte personal seguida por un nombre (esta regla ilustra el uso de la repetición en BNFs, cubriendo el caso de la gente que utiliza múltiples nombres y los nombres medios o las iniciales).
* Una dirección consiste de una especificación opcional del departamento, seguido de un número de casa, seguido por el nombre de la calle, seguido por un salto de línea (*end-of-line)*.
* Un apartado postal consiste de una ciudad, seguida por una coma, seguida por un código del estado (recuerde que es un ejemplo que ocurre en EE.UU.), seguido por un código postal y este seguido por un salto de línea (*end-of-line)*.

Observe que muchas cosas (tales como el formato de una parte personal, de una especificación del apartamento, o código postal) están dejadas sin especificar aquí. Si es necesario, pueden ser descritas usando reglas adicionales de BNF, o dejadas como [abstracción](https://es.wikipedia.org/wiki/Abstracci%C3%B3n_(inform%C3%A1tica)) si es inaplicable para el propósito actual.

Hay muchas variantes y extensiones de BNF, posiblemente conteniendo algunos o todos los comodines de [expresiones regulares](https://es.wikipedia.org/wiki/Expresi%C3%B3n_regular) como un "\*" o "+". El [Extended Backus-Naur form](https://es.wikipedia.org/wiki/BNF_extendido) (EBNF) es una variante común. De hecho el ejemplo anterior no es la forma pura inventada para el informe del ALGOL 60. La notación de los corchetes "[ ]" fue introducida algunos años más tarde en la definición de [PL/I](https://es.wikipedia.org/wiki/PL/I) de la [IBM](https://es.wikipedia.org/wiki/International_Business_Machines) pero ahora se reconoce universal. La [ABNF](https://es.wikipedia.org/w/index.php?title=Augmented_Backus-Naur_form&action=edit&redlink=1) es otra extensión usada comúnmente para describir [protocolos](https://es.wikipedia.org/wiki/Protocolo_de_comunicaciones) del [IETF](https://es.wikipedia.org/wiki/IETF).

Las [expresiones gramaticales de analizadores sintácticos](https://es.wikipedia.org/w/index.php?title=Expresi%C3%B3n_gramatical_de_analizador_sint%C3%A1ctico&action=edit&redlink=1) construidas en BNF y las notaciones de [expresión regular](https://es.wikipedia.org/wiki/Expresi%C3%B3n_regular) para formar una clase alternativa de la [gramática formal](https://es.wikipedia.org/wiki/Gram%C3%A1tica_formal), que es esencialmente [analítica](https://es.wikipedia.org/w/index.php?title=Gram%C3%A1tica_anal%C3%ADtica&action=edit&redlink=1) más que [generativa](https://es.wikipedia.org/wiki/Gram%C3%A1tica_generativa) en carácter.

Muchas especificaciones de BNF disponibles en línea tienen como propósito ser legibles a simple vista y no son especificaciones formales. Estas incluyen con frecuencia algunas de estas reglas sintácticas y extensiones:

* Elementos opcionales son presentados entre corchetes. Por ejemplo [<elemento-x>]
* Los elementos que se repiten 0 o más veces son presentados entre paréntesis de llave o terminados con un asterisco. Por ejemplo <palabra> ::= <letra> {<letra>}
* Los elementos que se repiten 1 o más veces son terminados con un '+'
* Los terminales pueden aparecer en negrillas y los no-terminales en texto normal en lugar de utilizar itálicas o paréntesis de ángulo
* Alternativas opcionales son separadas por el símbolo '|'
* Cuando se requiere agrupar varios elementos, se hace con paréntesis simples

**Notación formal para definir la sintaxis de un lenguaje:**

• Usada para especificar la mayoría de los lenguajes de programación

• Metasímbolos: ::= se define como | or { } repetición [ ] opcional

• Los terminales entre comillas y negrita, por ejemplo: 'if', '5'

Se agrega:

? Opcional

\* 0 a n veces

+ 1 a n veces

**Ejemplo anterior:**

con E-BNF numero ::= digito + ( '.' digito + ) ?

digito ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'

Las lenguas forman el terreno de la informática.

Los lenguajes de programación, las especificaciones de protocolo, los lenguajes de consulta, los formatos de archivo, los lenguajes de patrón, los diseños de memoria, los lenguajes formales, los archivos de configuración, los lenguajes de marcado, los lenguajes de formato y los meta-lenguajes dan forma a la forma en que computamos.

Entonces, ¿qué da forma a las lenguas?

Las gramáticas lo hacen.

Las gramáticas son el idioma de las lenguas.

Detrás de cada idioma, hay una gramática que determina su estructura.

Este artículo explica las gramáticas y las notaciones comunes para las gramáticas, como Backus-Naur Form (BNF), Extended Backus-Naur Form (EBNF) y las extensiones regulares de BNF.

Después de leer este artículo, podrás identificar e interpretar todas las notaciones comúnmente utilizadas para las gramáticas.

## Definiendo un idioma

Una gramática define un lenguaje.

En ciencias de la computación, el tipo más común de gramática es la gramática sin contexto, y estas gramáticas serán el enfoque principal de este artículo.

Las gramáticas libres de contexto tienen suficiente riqueza para describir la estructura sintáctica recursiva de muchos idiomas (aunque ciertamente no todos).

Discutiré las gramáticas más allá del contexto al final.

## Componentes de una gramática libre de contexto.

Un conjunto de reglas es el componente central de una gramática.

Cada regla tiene dos partes: (1) un nombre y (2) una expansión del nombre.

Por ejemplo, si estuviéramos creando una gramática para manejar texto en inglés, podríamos agregar una regla como:

*sustantivo-frase* puede expandirse en el *nombre del artículo* .

de lo cual podríamos deducir que "el perro" es una *frase sustantiva* .

O, si estuviéramos describiendo un lenguaje de programación, podríamos agregar una regla como:

*expresión* puede expandirse en *expresión* + *expresión*

Si estamos trabajando con gramáticas como objetos matemáticos, en lugar de escribir "puede expandirse a", simplemente escribiríamos : →

*nombre-frase* *artículo sustantivo expresión expresión expresión*→   
→ +

Como ejemplo, considérese la clásica gramática de expresión no ambigua:

e x p r → t e r m+e x p re x p r → t e r mt e r m → t e r m∗f a c t o rt e r m → f a c t o rf a c t o r → (e x p r)f a c t o r → c o n s tc o n s t → i n t e g e r

Entonces, ¿cómo sabemos que 3 \* 7es una expresión válida?

Porque:

*expr* puede expandirse a *término* ;   
que puede expandirse en *factor plazo* ; que puede expandirse en *factor factor* ; que puede expandirse en *factor de const* ; que puede expandirse en *const const* ; que puede expandirse en *const* ; que puede expandirse en .\*   
\*   
\*   
\*   
3 \*   
3 \* 7

## Notación de la forma Backus-Naur (BNF)

Al describir idiomas, la forma Backus-Naur (BNF) es una notación formal para codificar gramáticas destinadas al consumo humano.

Muchos lenguajes de programación, protocolos o formatos tienen una descripción BNF en su especificación.

Cada regla en la forma de Backus-Naur tiene la siguiente estructura:

n a m e ::= e x p a n s i o n

El símbolo ::=significa "se puede expandir en" y "se puede reemplazar con".

En algunos textos, un *nombre* también se denomina *símbolo no terminal* .

Cada *nombre* en la forma de Backus-Naur está rodeado por corchetes angulares, < >ya sea que aparezca en el lado izquierdo o derecho de la regla.

Una es una expresión que contiene símbolos terminales y símbolos no terminales, unidas por secuencia y elección. e x p a n s i o n

Un símbolo de terminal es un literal como ( "+"o "function") o una clase de literales (como integer).

Simplemente yuxtaponer expresiones indica secuenciación.

Una barra vertical |indica elección.

Por ejemplo, en BNF, la gramática de expresión clásica es:

<expr> :: = <term> "+" <expr> | <term> <term> :: = <factor> "\*" <term> | <factor> <factor> :: = "(" <expr> ")" | <const> <const> :: = entero

Naturalmente, podemos definir una gramática para las reglas en BNF:

r u l e → n a m e ::= e x p a n s i o n   
n a m e → < i d e n t i f i E r >   
e x p a n s i o n → e x p a n s i o n e x p a n s i o n   
e x p a n s i o n → e x p a n s i o n | e x p a n s i o n   
e x p a n s i o n → n a m e   
e x p a n s i o n → t e r m i n a l

Podríamos definir identificadores utilizando la expresión regular [-A-Za-z\_0-9]+.

Un terminal podría ser un literal entre comillas (como "+", "switch"o "<<=") o el nombre de una clase de literales (como integer).

El nombre de una clase de literales generalmente se define por otros medios, como una expresión regular o incluso una prosa.

## Notación BNF extendida (EBNF)

La forma extendida de Backus-Naur (EBNF) es una colección de extensiones de la forma de Backus-Naur.

No todos ellos son estrictamente un superconjunto, ya que algunos cambian la relación de regla de definición ::= a =, mientras que otros eliminan los paréntesis angulares de los no terminales.

Más importante que las pequeñas diferencias sintácticas entre las formas de EBNF son las operaciones adicionales que permite en las expansiones.

### Opción

En EBNF, los corchetes alrededor de una expansión , indica que esta expansión es opcional. [ *expansion* ]

Por ejemplo, la regla:

<term> :: = ["-"] <factor>

Permite negar factores.

### Repetición

En EBNF, las llaves indican que la expresión se puede repetir cero o más veces.

Por ejemplo, la regla:

<args> :: = <arg> {"," <arg>}

define una lista de argumentos separados por comas convencionales.

### Agrupamiento

Para indicar la precedencia, las gramáticas EBNF pueden usar paréntesis (), para definir explícitamente el orden de expansión.

Por ejemplo, la regla:

<expr> :: = <term> ("+" | "-") <expr>

Define una forma de expresión que permite tanto la suma como la resta.

### Concatenación

En algunas formas de EBNF, el ,operador denota explícitamente la concatenación, en lugar de confiar en la yuxtaposición.

## Notación de BNF aumentada (ABNF)

Las especificaciones de protocolo a menudo utilizan [la forma aumentada de Backus-Naur (ABNF)](http://en.wikipedia.org/wiki/Augmented_Backus%E2%80%93Naur_Form) .

Por ejemplo, [RFC 5322](http://tools.ietf.org/html/rfc5322#section-1.2) (correo electrónico), utiliza ABNF.

[RFC 5234](http://tools.ietf.org/html/rfc5234) define ABNF.

ABNF es similar a EBNF en principio, excepto que sus notaciones para la elección, la opción y la repetición difieren.

ABNF también brinda la capacidad de especificar exactamente valores específicos de bytes, detalle que importa en los protocolos.

En ABNF:

* elección es /; y
* opción utiliza corchetes: [ ]; y
* la repetición es *prefijo* \* ; y
* repetición *n*o más veces es *prefijo* *n*\* ; y
* La repetición *n*a los *m*tiempos es *prefijo* .*n*\**m*

EBNF se convierte en ABNF. { *expansion* }\*(*expansion*)

Aquí hay una definición de un formato de fecha y hora tomado de [RFC 5322](http://tools.ietf.org/html/rfc5322#section-1.2).

fecha-hora = [día de la semana ","] fecha hora [CFWS] day-of-week = ([FWS] day-name) / obs-day-of-week day-name = "Mon" / "Tue" / "Wed" / "Thu" / "Viernes" / "Sábado" / "Sol" fecha = día mes año day = ([FWS] 1 \* 2DIGIT FWS) / obs-day month = "Jan" / "Feb" / "Mar" / "Apr" / "Mayo" / "Junio" / "Julio" / "Agosto" / "Sep" / "Oct" / "Nov" / "Dec" año = (FWS 4 \* DIGIT FWS) / obs-year hora = zona de hora del día hora del día = hora ":" minuto [":" segundo] hora = 2DIGIT / hora obs. minuto = 2DIGIT / obs-minuto segundo = 2DIGIT / obs-segundo zone = (FWS ("+" / "-") 4DIGIT) / obs-zone

## Extensiones regulares a BNF

Es común encontrar operaciones [similares a expresiones regulares](http://matt.might.net/articles/sculpting-text/) dentro de las gramáticas.

Por ejemplo, la [especificación léxica de Python los](http://docs.python.org/3/reference/lexical_analysis.html) usa.

En estas gramáticas:

* postfix \*significa "repetido 0 o más veces"
* Postfix +significa "repetido 1 o más veces"
* postfix ?significa "0 o 1 veces"

La definición de literales de punto flotante en Python es un buen ejemplo de combinación de varias notaciones:

floatnumber :: = pointfloat | flotante exponentepointfloat :: = [intpart] fracción | intpart "."exponentfloat :: = (intpart | pointfloat) exponenteintpart :: = digit +fracción :: = "." dígito +exponente :: = ("e" | "E") ["+" | "-"] dígito +

No usa corchetes angulares alrededor de los nombres (como muchas notaciones EBNF y ABNF), pero sí usa ::=(como BNF). Mezcla operaciones regulares como +para la repetición no vacía con convenciones EBNF como [ ]para opción.

La [gramática de todo el lenguaje Python](http://docs.python.org/3/reference/grammar.html) utiliza una notación ligeramente diferente (pero aún así regular).

## Gramáticas en matematicas

Incluso cuando las gramáticas no son un objeto de estudio matemático en sí mismas, en los textos que tratan con estructuras matemáticas discretas, las gramáticas parecen definir nuevas notaciones y nuevas estructuras.

Para más información sobre esto, vea mi artículo sobre cómo [traducir las matemáticas a código](http://matt.might.net/articles/discrete-math-and-code/) .

## Más allá de las gramáticas libres de contexto.

Las expresiones regulares se ubican justo debajo de las gramáticas libres de contexto en el poder descriptivo: puede volver a escribir cualquier expresión regular en una gramática que represente los srings que coincidan con la expresión. Pero, lo contrario no es cierto: no toda gramática se puede convertir en una expresión regular equivalente.

Para ir más allá del poder expresivo de las gramáticas libres de contexto, es necesario permitir un grado de sensibilidad al contexto en la gramática.

La sensibilidad al contexto significa que los símbolos terminales también pueden aparecer en los lados izquierdos de las reglas.

Considera la siguiente gramática artificial:

<top> :: = <a> ")" <a> :: = "(" <exp> "(" <exp> ")" :: = 7

<top>puede expandirse hacia <a> ")";   
que puede expandirse hacia "(" <exp> ")";   
que puede expandirse en 7.

Si bien este cambio parece pequeño, genera gramáticas equivalentes a las máquinas de Turing en términos de los idiomas que pueden describir.

Al restringir las reglas de modo que el lado izquierdo tenga estrictamente menos símbolos que todas las expansiones a la derecha, las gramáticas sensibles al contexto son equivalentes a los autómatas de límite lineal (decidibles).

Aunque algunos lenguajes son sensibles al contexto, las gramáticas sensibles al contexto rara vez se usan para describir lenguajes de computadora.

Por ejemplo, C es ligeramente sensible al contexto debido a la forma en que maneja los identificadores y el tipo, pero esta sensibilidad al contexto se resuelve mediante una convención especial, en lugar de introducir la sensibilidad al contexto en la gramática.

Bibliografia:

* <http://lidis.usbcali.edu.co/Proyectos/matematicasDiscretas/2016-1/grupo1/guiasmodelos/Guia%20de%20la%20forma%20de%20backus%20naur.pdf>
* <https://es.wikipedia.org/wiki/Notaci%C3%B3n_de_Backus-Naur>
* <https://www.academia.edu/23198046/NOTACION_BACKUS_NAUR_BNF>
* <http://matt.might.net/articles/grammars-bnf-ebnf/>