VNIVERSIDAD NACIONAL AVPNOMA

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

AUTOMATIZACIÓN DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

T E S I S

QUE PARA OBTENER EL TITULO DE:
INGENIERO EN COMPUTACION

P R E S E N T A

ALEJANDRO JOSE GUADALUPE VAZQUEZ NAVA

Director de tesis:
Ing. Sergio Ruiz Palacios

MEXICO, D. F. 1990

AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Indice

4.4.- Sistema de nomenclatura de datos.

4.5.- Red semántica para presentar la relación existente entre la administración de datos, nomenclatura de datos y herramientas afines.

5.- SISTEMA DE NOMENCLATURA DE DATOS:

5.1.- Introducción.

5.2.- Descripción funcional.

5.2.1.- Requerimientos iniciales.

5.2.2.- Objetivo principal del sistema.

5.2.3.- Funciones del sistema.

5.2.4.- Productos del sistema.

5.3.- Fundamentos teóricos:

5.3.1.- Introducción.

5.3.2.- Introducción a lenguajes formales y autómatas.

5.3.3.- Lenguaje OF

5.3.4.- Revisión sintáctica del lenguaje español.

5.3.5.- Sintaxis del lenguaje de nomenclatura de datos.

5.3.6.- Autómata del sistema para la aplicación del lenguaje de nomenclatura de datos.

5.3.7.- Análisis morfológico en el sistema de nomenclatura de datos.

5.4.- Diseño lógico del sistema de nomenclatura de datos

5.4.1.- Red semántica del sistema de nomenclatura de datos.

5.4.2.- Descripción del autómata usado para la aplicación del lenguaje de nomenclatura de datos.

5.4.3.- Reglas para llevar a cabo el análisis sintáctico y semántico en la definición

del dato a nombrar.

5.4.4.- Descripcion del automata usado para el manejo de las ventanas del menu principal del sistema.

5.4.5.- Descripcion del automata usado para la deteccion y correccion de errores cometidos durante la especificación del texto de definición del dato a nombrar.

5.4.6.- Descripción del automata utilizado para la definicion de reglas de abreviacion que se usan para la generación de nombres primarios y nombres de programación.

5.4.7.- Descripción del autómata usado para controlar las ventanas que se usan en la fase de generacion de nombres primarios y nombres de programacion.

5.4.8.- Descripcion del automata usado para la definición de la estructura de los documentos de reportes generados por el sistema.

5.4.9.- Descripción del autómata encargado de realizar el análisis morfologico de las palabras que forman parte de la definicion del dato. (análisis de sustantivos y adjetivos).

5.4.10.- Relaciones existentes entre los diferentes autómatas del sistema.

5.5.- Diseño físico:

5.5.1.- Manejo del ambiente orientado a ventanas.

5.5.2.- Manejo de estructuras de datos dinámicas.

5.5.3.- Manejo de archivos.

5.5.4.- Manejo del teclado.

5.5.5.- Manejo de procesos de validación y seguridad del sistema.

5.6.- Construcción del sistema:

5.6.1.- Carta estructurada del sistema.

5.6.2.- Descripción de las principales rutinas del sistema.

1.- INTRODUCCION:

Se ha observado que los datos de los sistemas se han definido de acuerdo a un uso particular y especifico para su estructura funcional, dejando aparte un enfoque estandarizado que permita identificar a los datos comunes a distintos sistemas, y de esta forma compartirlos. La estructura de los datos para los registros de archivos está diseñada bajo un esquema de demanda de uso, a fin de minimizar el acceso a disco; debido a esto, la mayoría de los sistemas tienen un "archivo maestro" que almacena la información más demandada.

De acuerdo a lo anterior, se ha observado que existe la siguiente problemática:

1) Existe redundancia innecesaria de datos comunes a diferentes sistemas.

2) Existen datos comunes a diferentes sistemas, pero que en cada uno están definidos con diferentes longitudes y valores.

3) La nomenclatura de los campos está asociada a los archivos, no existe una estandarización que permita que un dato sea nombrado en forma única e independiente de programas y formas de almacenamiento.

4) La documentación de descripciones de datos almacenados en archivos, es obsoleta o se encuentra desactualizada.

Ante esta situación, se consideran varias etapas en la evolución de los departamentos de sistemas de las instituciones. Estas etapas muestran las siguientes características:

1) Inicio y 2) Motivación

En estas etapas, los esfuerzos en el desarrollo de aplicaciones se concentran en la automatizacion y soporte, realizándose la planeación y control internos para administrar a los centros de cómputo. En estas etapas iniciales, no existen datos compartidos entre distintas aplicaciones. Cada aplicación tiene sus propios archivos independientes. Existe gran proliferación de archivos, con alto nivel de duplicación, lo que leva a altos costos de mantenimiento.

3) Control:

En esta fase, las aplicaciones empiezan a considerar la generación y uso de los datos, el departamento de sistemas se convierte en el custodio de los datos. Empiezan a diseñarse e implantarse bases de datos, pero aún sin llegar a compartir datos comunes; las bases de datos generadas en esta etapa existen ligadas a las aplicaciones.

Los usuarios finales empiezan a involucrarse con el manejo de los datos, siendo responsables de su calidad y valores permitidos.

4) Integración:

En esta etapa, se presenta un punto de transición en la evolución: Se inicia una planeación para administrar el recurso de los datos. Se establecen comités para establecer la administración de datos a nivel de área de negocio y corporativa. Empiezan entonces a crearse bases de datos independientes de aplicaciones específicas. Se requiere en esta etapa del análisis y modelado de datos para identificar a los datos comunes a diferentes áreas del negocio de la institución.

5) Administración de datos y 6) Madurez:

En estas etapas, se alcanza el balance entre datos y sistemas comunes. En la institución se implanta el concepto de administración de datos con sus procedimientos y responsabilidades. La planeación de datos se vuelve más cuidadosa y se hace con la participación de los usuarios finales. Se siguen creando bases de datos temáticas, pero a partir de ellas se extraerán datos para crear bases de datos para dar soporte a la toma de decisiones. El recurso que representan los datos es explotado en todos los niveles de la institución.

A lo largo de esta evolución, el ciclo de vida de los sistemas amplía su panorama. Ya deja el tradicional esquema de análisis, diseño, construcción y mantenimiento. El ciclo se amplía y se basa ahora en una planeación a largo plazo, la cual prioriza a los proyectos y da como resultado una arquitectura de sistemas la cual se usará como base para el desarrollo de los sistemas. En ella se identificarán y usarán datos comunes, rutinas comunes, software estándar y diseños reutilizables. En base a esta arquitectura, el desarrollo se simplificará y aumentará su calidad.

Es evidente que esta evolución representa el esfuerzo de hallar datos comunes para mejorar la calidad de la información y de esta forma desarrollar sistemas cada vez más confiables. Por esta razón, aparece la necesidad de la función de administración de datos.

La presente tesis tiene como objetivo presentar un panorama sobre la administración de datos, sus funciones y beneficios, así como herramientas asociadas. Entre sus funciones, será mostrada la nomenclatura de datos, la cual es un elemento clave que ayuda a establecer las bases para la identificación de los datos comunes a diferentes áreas del negocio de la institución. Asimismo, se mostrará que las herramientas de software constituyen un factor de éxito para

la generación y aplicación de los estándares de la administración de datos. Se mostrará la conveniencia de la generación de una herramienta que automatice la nomenclatura de datos. En los ultimos capitulos se describen los conceptos usados para el diseño del sistema, asi como las caracteristicas fisicas del mismo.

2.- ADMINISTRACION DE DATOS:

2.1.- Introducción:

La administración de datos representa una fase en la evolución del quehacer en el área de sistemas. Esta evolución lleva a considerar a los datos como uno de los recursos más importantes de la institución, y por lo tanto, requiere administrarse correctamente. Lo anterior implica la necesidad de una planeación orientada a datos, una serie de actividades en el desarrollo de sistemas que estén orientadas al diseño de bases de datos, así como de la creación, implantación y uso de estándares que permitan revisar la calidad de los sistemas.

En el presente capitulo se explican los principales objetivos, funciones y beneficios de la administración de datos, así como la estructura organizacional requerida para llevar a cabo dichas funciones.

2.2.- Definición:

La administración de datos es una disciplina que se encarga de mejorar la calidad de la información. Lo anterior implica que debe facilitar la **planeación, obtención, clasificación, almacenamiento, control y seguridad de los datos** a partir de los cuales se genera la información que se requiere para el buen funcionamiento de una institución.

2.3.- Objetivos principales:

Los objetivos primarios de la administración de datos son los siguientes:

1) Identificar, definir, nombrar y administrar a los datos corporativos que se requieren para el funcionamiento óptimo de la institucion, considerandolos como uno de sus recursos más importantes.
2) Definir y manejar modelos de datos que faciliten el desarrollo de nuevos sistemas.
3) Organizar a los datos de forma tal que puedan ser compartidos e independientes de las aplicaciones que requieran de ellos. El compartir los datos apoya el desarrollo de nuevas aplicaciones y el uso de estándares. La independencia de datos permite una mayor precisión y consistencia de la información, dando a los sistemas una mayor flexibilidad.

2.4.- Alcances:

La administración de datos impacta a toda la estructura del área de sistemas de la institución, de manera tal que se incremente la disponibilidad y uso de los datos en forma global y consistente; de esta forma se proporciona un apoyo sólido y confiable para la toma de decisiones a niveles directivos y gerenciales.

2.5.- Beneficios:

Entre los beneficios más significativos y de mayor impacto se pueden contar los siguientes:

1) Se incrementa la calidad de la información de la institución, al incrementar la precisión y la confiabilidad de los datos.
2) Permite la estandarización en la nomenclatura y uso de los datos.
3) Apoya la integración de una arquitectura de datos corporativa.
4) Permite manejar los datos en forma más consistente y y oportuna en toda la institución al identificar y establecer relaciones entre los datos.
5) Favorece la independencia de datos, pues no estarán - ligados a las aplicaciones en lo referente a la forma de acceso, ni a la organización física. El grado de independencia de datos que se alcance, dependerá de las facilidades que brinde el DBMS que se use.
6) Se reducen los costos de mantenimiento, al tener redundancia mínima de datos y código, ya que al tener control sobre su dispersión, se facilita la localización de los datos y de los programas que los usan.

2.6.- Funciones:

Las funciones principales de la administración de datos son las siguientes:

Planeación, Diseño lógico de bases de datos, Calidad de datos, Control del esquema de las bases de datos y su seguridad.

A continuación se explica cada una de estas funciones:

a) Planeación:

a.1) Metodología de planeación:

Es función de la administración de datos evaluar metodologías orientadas a la planeación estratégica de datos y a modelos de información. Por ejemplo: BSP (Business Systems Planning) o CSF (Critical Success Factors).

Una vez seleccionada la metodología, la administración de datos deberá desarrollar un plan de implantación de ésta en la institución. En caso de que la metodología ya exista en la institución, la labor de la administración de datos se reduce a modificarla o expandirla para incluir tareas que estén orientadas a datos.

a.2) Planeación estratégica de datos:

Esta función consiste en desarrollar una estrategia

global que satisfaga las necesidades de información de la institución a largo plazo.

Estos planes se basan en:

- El análisis de datos básicos para el desarrollo de los modelos de información.
- En la recopilación de las necesidades de información de la institución.

Esta planeación estratégica deberá dar como resultado un plan que indique sistemas de información que deberán desarrollarse bajo ciertas prioridades y las bases de datos corporativas que también deberán existir para dar soporte a estos sistemas; de esta forma se compartirán datos comunes a diferentes sistemas que pudiesen pertenecer a diversas áreas del negocio de la institución.

a.3) Modelado de información:

En esta función, la administración de datos desarrollará modelos de información corporativo y por área de negocio de la institución, de forma tal que dichos modelos apoyen el análisis y la comprensión de los datos básicos que se usen para satisfacer las necesidades de información de la institución.

b) Diseño lógico:

b.1) Diseño lógico de bases de datos:

Esta funcion de la administración de datos consiste en descomponer y afinar las entidades identificadas en el modelo de información. También incluye a la - identificación de relaciones entre entidades, reglas de integridad, llaves lógicas y atributos. Esta afinación da como resultado los modelos de datos de las bases de datos que se desarrollen de acuerdo a las prioridades identificadas por la planeación estratégica.

Una vez identificados los atributos y relaciones existentes entre las entidades, es deber del administrador de datos aplicar el proceso de normalización de la base de datos. De esta forma, se evitarán anomalías en los procedimientos de inserción, actualización y supresión de registros en la bases de datos real. Normalmente se asociaba esta función al administrador de bases de datos, sin embargo es el administrador de datos quien puede determinar (a través del conocimiento de los datos y su significado) la existencia de las dependencias funcionales existentes entre los distintos atributos de las entidades. El conocimiento de estas dependencias funcionales es fundamental para llevar a cabo el proceso de

normalizacion de bases de datos.

En esta funcion también se contempla la nomenclatura de datos y la aplicacion de sus estándares al nombrar a los datos de la institución.

b.2) Conversión lógico a físico:

Una vez obtenido el diseño lógico de la base de datos, es necesario considerar las restricciones del ambiente físico en el que habrán de desarrollarse los sistemas de información: Estas restricciones se refieren al tipo de estructuras físicas de datos soportadas por los D.B.M.S.(Sistemas Manejadores de Bases de Datos), nomenclatura interna de sistemas, etc.

En esta conversión, se deberá asegurar que la integridad de los datos y la calidad de la información están adentro de los límites de aceptación convenidos con los usuarios.

b.3) Pruebas de la base de datos.

Una vez que la base de datos ha sido creada a través del DBMS, el administrador de datos tiene la responsabilidad de probar que la base de datos es capaz dar soporte a los requerimientos de información previstos en el desarrollo de la misma. Para

ello, deberá considerar las transacciones típicas que habrán de afectar a la base de datos. Estas pruebas tienen como objeto revisar que las relaciones identificadas en el modelo de datos son correctas, así como los atributos identificados en cada entidad. A través de las especificaciones de las reglas de integridad, el administrador de datos también revisará que los procedimientos de validación de datos son correctos.

Es importante notar que en estas pruebas, el administrador de datos no evalúa la eficiencia de la base de datos, ni tampoco las estructuras físicas de datos que se usaron para construirla. Su misión es evaluar que la base de datos brinde la información para la cual fue diseñada, sin importar su eficiencia.

c) Calidad de datos:

c.1) Estándares y procedimientos:

La administración de datos tiene la función de dictar los estándares y procedimientos que asegurarán la calidad de la información manejada por los sistemas.

Estos procedimientos y estándares se aplicarán tanto a la planeación, diseño conceptual, nomenclatura

de datos, así como al control de calidad durante el desarrollo de los sistemas.

De este modo, pueden contemplarse a sistemas que automaticen a los estándares de nomenclatura de datos.

c.2) Control de cambios:

En esta función, la administración de datos establece y monitorea los procedimientos de control de cambios para definiciones de datos para asegurar que todos los cambios sean hechos a tiempo y que sean comunicados y coordinados bajo el consentimiento de las áreas afectadas por dichos cambios.

d) Control del esquema de las bases de datos y seguridad:

Una vez que se ha definido la estructura lógica y física de la base de datos, se debe tener un control sobre los subesquemas de la base de datos y los usuarios (personas o programas) que tienen acceso a dichos subesquemas. Estos subesquemas estarán formados por porciones de la base de datos, conteniendo a ciertos archivos, campos y relaciones establecidas a través de mecanismos proporcionados por el DBMS. La administración de datos tendrá la responsabilidad de llevar el control de los campos

definidos en cada uno de los archivos que forman parte de cada subesquema y asegurar que no existen redundancias innecesarias en la base de datos. Asimismo, deberá asegurar que diferentes campos de archivos que estén asociados a un dato en particular, estarán definidos con las mismas características físicas, de forma tal que puedan ser compartidos entre diferentes programas.

Este control también deberá permitir al administrador de datos tener información acerca de los programas que hacen uso de determinado dato que está contemplado en cierta base de datos. Dicha información permitirá identificar el impacto que pueda representar una posible reestructuración de la base de datos; es decir, si es necesario hacer una recodificación de programas, esta información deberá indicar: programas a recodificar, líneas a cambiar, nuevos campos de la base de datos a contemplar, campos a eliminar, etc.

Al tener control sobre los subesquemas de la base de datos y conocer a los usuarios que tienen acceso a dichos subesquemas, se establece un mecanismo de seguridad en el uso de los datos. El administrador de datos debe tener información acerca de los usuarios y los subesquemas a los cuales tiene acceso, así como el tipo de operaciones a las que tiene autorización de

ejecutar en la base de datos.

Para llevar a cabo estos controles, el administrador de datos tiene a su disposición una herramienta conocida como "Diccionario de Datos", la cual se explica en el capitulo 4.

2.7.- Estructura organizacional:

La administración de datos debe organizarse de acuerdo a la estructura organizacional de la institución. De esta forma, puede pensarse en varias posibilidades:

a) Estructura organizacional por áreas del negocio:

Cuando la estructura organizacional del área de sistemas está de acuerdo a las áreas del negocio de la institución (por ejemplo que se divida de acuerdo a las diferentes bancas de una institución bancaria) puede considerarse que existirá un equipo de administración de datos por cada área del negocio. En este caso, deberá existir otro equipo, aparte de los anteriores, el cual deberá estar encargado de la administración de datos corporativa, llevando a cabo la función de planeación estratégica de datos. Estos dos tipos de grupos realizan funciones operativas dentro de la administración de datos, sin embargo, deberá existir un tercer grupo. Este grupo tendrá la responsabilidad de

investigar, apoyar y coordinar las funciones y herramientas asociadas a la administración de datos.

Bajo este esquema de organizacion, las responsabilidades y funciones asociadas a cada equipo de administracion de datos son las siguientes:

Equipo de administración de datos corporativo:

Este equipo está encargado de llevar a cabo la planeación estratégica de datos bajo cierta metodología (como BSP ó CSF). Este equipo tendrá la responsabilidad de llevar un analisis de la información requerida a nivel institucional y en un esquema macroscópico, generar un modelo de informacion que muestre los datos organizados en forma de entidades y relaciones. Este modelo de información será la guia para determinar las bases de datos temáticas comunes a la institución en sus diferente área del negocio. Debido a lo anterior, deberá existir una relación muy fuerte entre este grupo de trabajo y los demás grupos de administracion de datos pertenecientes a las diferentes áreas del negocio.
Las entidades, relaciones y ciertos atributos comunes a la institución serán administrados y dados a conocer a los grupos de administración de datos de cada una de las áreas del negocio, de forma tal que sean respetados los nombres de los datos y se consideren sus definiciones durante el desarrollo de las diferentes bases de

datos. Estas definiciones y nombres serán determinados por cada grupo de trabajo y deberán llegar a un concenso general, de forma tal que sean adoptados fácilmente en cada área.

Equipos de administración de datos por área del negocio:

Estos equipos de administración de datos, se encargarán de la función de diseño lógico de las bases de datos, siguiendo las directrices trazadas por la planeacion estratégica de datos. Los modelos de datos que se obtengan, deberán ser compatibles con el modelo de información obtenido en la planeación estratégica de datos. Estos modelos de datos reflejarán las entidades, relaciones y atributos de entidades que servirán para manejar a los datos de estas bases de datos. Asimismo, asociados a estos modelos de datos existirán reglas de integridad que indicarán los mecanismos de validación que deberán seguirse para asegurar el buen estado de estas bases de datos al momento de hacer actualizaciones en ellas.

En el desarrollo de estas bases de datos, cada grupo de administración de datos se encargará de llevar a cabo la conversión lógico a físico, aplicando los estándares de nomenclatura de datos implantados en la institución.

Equipo de investigación de administración de datos:
Este equipo es responsable de investigar, coordinar e impulsar la administracion de datos en las diferentes áreas en las que se estructure el departamento de sistemas de la institucion. Asimismo, este equipo será responsable de la evaluacion y recomendación de herramientas asociadas a la administración de datos.

b) Estructura organizacional centralizada:

En el caso de que se trate de una organización de sistemas central, la administración de datos será responsabilidad de un sólo grupo. Este grupo estará encargado de todas las funciones de la administración de datos y será responsable de dictar y aplicar los estándares de datos. También estará a cargo de la investigación, evaluación y recomendación de herramientas asociadas a la administración datos.

2.8.- Productos:

Entre los productos más importantes que genera la administración de datos, se encuentran los siguientes:

a) Estándares de administración de datos, que incluyen a los siguientes:
   a.1) Estándares de nomenclatura de datos.
   a.2) Estándares de modelado de información.
   a.3) Estándares de datos.

b) Modelo de información corporativo y modelos de datos de cada área del negocio.

c) Políticas y procedimientos operativos respecto a:

   c.1) Uso de herramientas destinadas al apoyo de las funciones de la administración de datos:

      - Planeación.
      - Diseño lógico.
      - Establecimiento y aplicación de estándares.

   c.2) Entradas y actualización de metadatos.

   c.3) Seguridad de diccionarios de datos.

   c.4) Definición de responsabilidades de usuarios.

d) Diseño o adecuación del modelo lógico del diccionario de datos en el que habrán de almacenarse los metadatos.

e) Inventario de datos.

f) Inventario de programas.

3.- ESTANDARES DE ADMINISTRACION DE DATOS:

3.1.- Introducción:

Como se estableció en el capítulo anterior, la administración de datos define una serie de funciones y actividades orientadas a la identificación, almacenamiento y control de los principales datos de la institución. Al contar con una estructura organizacional encargada de estas responsabilidades, es evidente que diferentes personas estarán involucradas en diferentes proyectos y en diferentes actividades de la administración de datos. Lo anterior implica la posibilidad de que proliferen una serie de técnicas (equivalentes en su propósito, pero distintas en su forma y presentación de resultados) que dificulten compartir información sobre los proyectos, y por tanto, identificar datos comunes a diferentes áreas del negocio. Esto daría como resultado una baja calidad en los productos de los proyectos de sistemas.

Por esta razón, la administración de datos requiere establecer una serie de estándares o normas que rijan las funciones, actividades y técnicas de la misma administración de datos.

En este capítulo se presentan los principales objetivos, beneficios y estándares más importantes para la administración de datos.

3.2.- Objetivos:

El objetivo que se persigue al fijar estándares de administración de datos es incrementar la productivad en el área de sistemas y las áreas de usuarios finales a través de:

a) Reforzar la consistencia en el uso de los datos:

Con estándares en la nomenclatura de datos, se obtienen nombres precisos y comprensibles que facilitan la detección y reducción de ocurrencias de diferentes datos bautizados con el mismo nombre (homónimos),o de varios nombres oficiales para el mismo dato (sinónimos) o varios nombres no oficiales para el mismo dato. Con esto, se minimizan la redundancia y el uso duplicado de datos usados en los sistemas de información de la institución.

También se contempla la estandarización de longitudes y tipos de datos, de forma tal que en los programas de aplicación no tengan que programarse rutinas de conversión de formatos, longitudes y tipos.

b) Incrementar la precisión y disponibilidad de los datos:

Los estándares de administración de datos mejoran la forma de identificar y clasificar los datos de la institución. Al proporcionar nombres significativos a los datos, se mejora la precisión en la búsqueda de los datos. También se facilita la asociación y el

acceso a datos con características y usos similares y por lo tanto se minimiza la posibilidad de que se presenten inconsistencias e irregularidades al actualizar a bases de datos complejas.

3.3.- Políticas:

Para que la definición y adopción de estándares de administración de datos tengan éxito, se recomiendan las siguientes políticas:

a) Los niveles de dirección apoyarán y fomentarán el uso de los estándares.

b) Los estándares no pueden ser retroactivos, ni se pueden aplicar a sistemas que existan con anterioridad a los estándares.

c) Los estándares deberán ser prácticos, de fácil utilización y adopción. Se fomentará y procurará la utilización de herramientas automatizadas de apoyo. (P.ej. diccionarios de datos, sistema de nomenclatura de datos, sistemas para control del uso de estándares, etc).

3.4.- Beneficios:

A través de los estándares de administración de datos, se obtienen los siguientes beneficios:

a) Incremento en la calidad de la documentación: Los estándares de administración de datos requieren definiciones significativas y descrip-

tivas de los datos.

Sin la suficiente documentación, los sistemas se encuentran llenos de redundancias e inconsistencias, lo cual contribuye a errores de incompatibilidad de datos durante el desarrollo y la operación de los sistemas de información.

Los estándares de nomenclatura de datos permiten que los nombres de los datos sean más significativos y por lo tanto autodocumentables.

A través de diferentes medios, los datos se definen, nombran y documentan: Sistema de nomenclatura de datos, diccionarios de datos, etc. Con lo anterior, se logra incrementar la consistencia de los datos.

b) Reducción en el costo de mantenimiento de sistemas:

La creación de modelos de información, bases de datos y el uso de diccionarios de datos, permiten el manejo, detección y corrección de errores en forma rápida y efectiva, reduciendo los costos de mantenimiento de sistemas de información.

Al aplicar estándares para el uso eficiente de los datos, se minimizan los recursos usados en la transformación del formato físico de los datos.

c) Incremento en la modularidad y flexibilidad de

los sistemas de información:

Este beneficio se obtiene al minimizar la redundancia de datos y permitir que los datos sean compartidos entre diferentes sistemas, a través del uso de rutinas comunes.

3.5.- Estándares de nomenclatura de datos:

3.5.1.- Objetivos de la nomenclatura de datos:

Los objetivos de la nomenclatura de datos, son los siguientes:

- Identificar, definir, describir, clasificar y establecer la unicidad de los datos de la institución.
- Satisfacer requerimientos de lenguajes de programación y software con nombres estándares.
- Facilitar el entendimiento de cuáles campos son redundantes, cuáles son sinónimos, homonimos y alias.

El objetivo primordial consiste en:

Crear, para cada dato, una definición única y que sea entendida por las personas que trabajan en la institución. A partir de estas definiciones se nombrarán en forma única a los datos de la institución y puestos a disposición del personal de sistemas y usuarios finales a través del o los diccionarios de datos.

Estas definiciones facilitan el entendimiento de cuáles campos contienen datos redundantes y cuáles son homónimos (diferentes datos con el mismo nombre), sinónimos

(mismo dato con 2 o más nombres oficiales), y alias (nombres no oficiales para un mismo dato).

3.5.2.- Componentes de los estándares de nomenclatura de datos:

A partir de la **definición** de cada dato, se creará el correspondiente **nombre primario**, que será la llave para tener acceso a los metadatos que se encuentran almacenados en el diccionario de datos. Asimismo, también la definición del dato se usará para crear los **nombres de programación**, que corresponden a los nombres de los campos usados para representar al dato en forma física en los ambientes de programación de sistemas. Tanto el **nombre primario** como el **nombre de programación** se construirán a partir de las **abreviaturas** de palabras claves contenidas en la definición del dato. Dichas abreviaturas deberán regirse en base a una serie de **reglas de abreviación** dictadas por los estándares de nomenclatura de datos.

Es lógico pensar que la **definición** de cada dato requiere contener ciertas **palabras claves** que permitirán construir el nombre primario y los nombres de programación. Estas palabras claves, se clasifican en tres categorías:

- **Palabras clase**
- **Palabras entidad**
- **Palabras modificadoras**

Palabra Clase: Es la palabra que describe en forma explícita al dato, de acuerdo a su uso en los procesos del negocio. Específicamente, la palabra clase contesta a la pregunta ¿Qué es?

Algunos ejemplos de palabras clase son los siguientes:

nombre, número, porcentaje, fecha

Existen palabras clase cuyo significado hace que no se requiera de palabras modificadoras en la definición del dato. Dichas palabras clase se conocen como **"palabras clase de modificador opcional"**.

Por ejemplo: Nombre

Esta palabra puede usarse en la definicion del dato: "Nombre del cliente". En este caso, la informacion está completa, pues se contesta a las preguntas: ¿Qué es?, ¿A quién se refiere?.

Por otro lado, existen las palabras clase cuya información requiere enriquecerse con un modificador adicional. Dichas palabras clase se conocen como **"palabras clase de modificador forzoso"**.

Por ejemplo: Fecha

Esta palabra clase requiere de modificador, pues al usarla en una definición como: "fecha de empleado", surgen las preguntas sobre si esta fecha es de nacimiento, o de contratación. Por esta razón, la palabra clase "fecha"

requiere que se use un modificador que enriquezca la información de la definición del dato a nombrar.

Palabra Entidad: Es un sustantivo que generalmente identifica a los sujetos del negocio acerca de los cuales se quiere guardar información. La palabra entidad contesta a la pregunta ¿De quién?.

Las palabras entidad deben coincidir con las entidades identificadas en los modelos de información y modelos de datos corporativo y de las diversas áreas del negocio de la institución.

Palabras modificadoras: Son palabras que complementan la definición del dato. Ayudan a hacer más clara la definición del dato. Dependiendo del tipo de la palabra clase usada en la definición del dato a nombrar, la presencia de un modificador será forzosa u opcional.

No se consideran como palabras claves, a aquellas como: artículos, preposiciones, letras sueltas, etc.

Para la **construcción de los nombres primarios y de programación,** es necesario considerar una serie de **reglas** que permitan la **creación de abreviaturas.** A continuación se muestra la lista de **reglas de abreviación** que deben utilizarse, así como la explicación de cada una de ellas.

a) Orden de palabras abreviadas en los nombres
b) # máximo de caracteres por palabra abreviada

c) # mínimo de caracteres por palabra abreviada

d) # máximo de palabras por nombre

e) Sustitución del gramema de género

f) Sustitución del gramema de número

g) Sustitución de la letra "ñ"

h) Eliminación de la letra inicial

i) Eliminación de la letra final

j) Eliminación de vocales internas

k) Eliminación de consonantes iguales

l) Eliminación de caracteres numéricos

m) Uso de abreviaciones especiales

n) Caracter conector de palabras abreviadas

a) Orden de palabras en los nombres :

Al momento de generar el nombre primario o nombre de programación, las palabras claves de la definición del dato a nombrar pueden tomar cualquiera de las siguientes secuencias:

1: Clase, Entidad y Modificadores

2: Clase, 1er Modificador, Entidad y Modificadores

3: Entidad, Clase y Modificadores

4: Entidad, 1er Modificador, Clase y Modificadores

5: 1er Modificador, Clase, Entidad y Modificadores

6: 1er Modificador, Entidad, Clase y Modificadores

7: Mismo que en la descripción del dato

b) **# máximo de caracteres por palabra abreviada :**
Cada palabra clave que se abrevie deberá obedecer a un máximo de caracteres que la compongan.

c) **# mínimo de caracteres por palabra abreviada :**
Cada palabra clave que se abrevie deberá obedecer a un mínimo de caracteres que la compongan.

d) **# máximo de palabras por nombre :**
Cada nombre primario o de programación deberá obedecer a un número máximo de palabras abreviadas que lo compongan. Para el número mínimo, se define que tanto los nombres primarios como los de programación estarán formados por:

2 palabras : la palabra clase y entidad en el caso de que que se trate de que la palabra clase sea de tipo de modificador opcional.

3 palabras : la palabra clase, modificador forzoso y la palabra entidad, en el caso de que se trate de que la palabra clase sea de tipo de modificador forzoso.

e) **Sustitución del gramema de género :**
Al momento de generar los nombres, deberá revisarse que exista coincidencia de los gramemas de género entre las palabras claves a abreviar. En este caso, puede

adoptarse cualquiera de las siguientes acciones:

1: Sustitucion de masculino a femenino

Las palabras escritas en masculino son cambiadas en su gramema para quedar en femenino.

2: Sustitucion de femenino a masculino

Las palabras escritas en femenino son cambiadas en su gramema para quedar en masculino.

3: No hay sustitución

Las palabras no son cambiadas en su gramema de género.

f) Sustitución del gramema de número :

Al momento de generar los nombres, deberá revisarse que exista coincidencia de los gramemas de número entre las palabras claves a abreviar. En este caso, puede adoptarse cualquiera de las siguientes acciones:

1: Sustitución de singular a plural

Las palabras escritas en singular son cambiadas a plural.

2: Sustitución de plural a singular

Las palabras escritas en plural son cambiadas en su gramema de número para quedar en singular.

3: No hay sustitución

En este caso, no se sustituyen los gramemas de número.

g) **Sustitución de la letra "ñ" :**

Es bien conocido que los ambientes de programación no dan soporte a ciertos caracteres del lenguaje español, especialmente a la letra "Ñ". Por esta razón, normalmente la letra "ñ" se sustituye por cualquier otro caracter. Es deber del administrador de datos indicar el caracter que sustituirá a la letra "ñ" en los nombres de los datos.

Las siguientes reglas se refieren a la serie de acciones a tomar para eliminar letras de las palabras que se desean abreviar:

h) **Eliminación de la letra inicial :**

A través de esta regla, se eliminará (o no) a la primera letra de las palabras claves a abreviar. Esta acción se efectuará dependiendo del estado en el que se encuentre esta regla:

1: Regla de abreviación activa
2: Regla de abreviación inactiva

i) **Eliminación de la letra final :**

A través de esta regla, se eliminará (o no) a la última letra de las palabras claves a abreviar. Esta acción se efectuará dependiendo del estado en el que se encuentre

esta regla:

1: Regla de abreviacion activa
2: Regla de abreviación inactiva

j) Eliminación de vocales internas :

A través de esta regla, se eliminarán (o no) a las vocales internas de las palabras claves a abreviar. Esta acción se efectuará dependiendo del estado en el que se encuentre esta regla:

1: Regla de abreviación activa
2: Regla de abreviación inactiva

k) Eliminación de consonantes iguales :

Las reglas anteriores pueden generar que en la abreviación de las palabras queden letras consonantes adyacentes iguales. Esta regla se encarga de eliminar a dichas consonantes. Esta acción se efectuará dependiendo del estado en el que se encuentre esta regla:

1: Regla de abreviación activa
2: Regla de abreviación inactiva

l) Eliminación de caracteres numéricos :

A través de esta regla, se eliminarán (o no) a los

caracteres numéricos que existan en las palabras claves a abreviar. Esta acción se efectuará dependiendo del estado en el que se encuentre esta regla:

1: Regla de abreviación activa
2: Regla de abreviación inactiva

m) Uso de abreviaturas especiales :

Es normal que existan palabras que tengan una abreviatura convencional, aceptada por la institución, y que sin embargo no se haya construido a partir de las reglas de abreviación establecidas por la administración de datos. Por esta razón, se desea que puedan usarse estas abreviaturas especiales al momento de generar los nombres de los datos. Estas abreviaturas especiales sólo existirán para las palabras clase y para palabras entidad; las abreviaturas de las palabras modificadoras siempre se construirán de acuerdo a las reglas de abreviación. Al momento de generar las abreviaturas, si esta regla se encuentra activa, se usará la abreviatura especial de la palabra y se omitirán las demás reglas de abreviación (reglas g,h,i,j,k,l). Si esta regla se encuentra inactiva, entonces la abreviación de cada palabra se hará de acuerdo a las demás reglas de abreviación.

n) **Caracter conector de palabras abreviadas :**

Una vez que se han abreviado todas y cada una de las palabras claves, de acuerdo al estado de las reglas de abreviación descritas anteriormente, es necesario concatenarlas para formar al nombre primario o nombre de programacion. Para ello, se utilizarán caracteres especiales para unir a las palabras abreviadas. Puede considerarse a cualquiera de las siguientes opciones:

1: Guión -

2: Subguión _

3: Punto .

4: Caracter especial definido por el administrador de datos

3.5.3.- Consideraciones sobre la nomenclatura de datos:

Una vez que se ha definido y nombrado a un dato y que se ha almacenado en el diccionario de datos, sus ocurrencias (campos en archivos, variables en programas) se deben reconocer y asociar con el dato al que se refieren.

La nomenclatura de datos, incluyendo el nombre primario y los nombres de programación, debe ser responsabilidad de pocas personas, las cuales trabajan en la administración de datos. Esto último se debe a que con pocas personas, es posible mantener la consistencia en la aplicación de nomenclatura de datos. Debido a que los datos se definen una sola vez (a menos que se redefinan), el trabajo inicial es muy grande; sin embargo, con el tiempo este trabajo disminuirá hasta que sólo se tenga que definir a los nuevos datos.

Es responsabilidad del equipo de administración de datos el comparar la definición de un datos que se está creando, con las definiciones ya existentes. Si no se encuentra una definición igual o equivalente, en tal caso el responsable creará el nombre primario y los nombres de programación y los almacenará en un acervo de nombres.

3.6.- Estándares de datos:

La estandarizacion de datos se divide en 2 puntos:

a) estandarización de características físicas de los datos.

b) estandarización de valores de datos.

La estandarización de las características físicas de los datos, se refiere a los siguientes puntos:

- Definición de la longitud de los campos y variables que corresponden a los datos identificados por la administracion de datos.
  P.ej. el dato :"Nom-emp" (Nombre del empleado) debe estar contenido en un sólo campo y con una longitud máxima de 40 caracteres.

- Definición del tipo de caracteres a usarse.
  P.ej. el dato: "Nom-emp" debe contener caracteres alfabèticos, pero no numericos.

- Definición de la sintaxis de los datos:
  P.ej. el dato "Nom-emp" debe contener en primer lugar al apellido paterno del empleado, luego el apellido materno y por último el o los nombres del empleado.

- Definición de los estándares de almacenamiento físico. P.ej. Todos los datos de tipo numérico deben almacenarse en decimal empacado.

La estandarización de los valores de datos, tiene que ver con lo siguiente:

Legalizar los valores o códigos válidos que pueden usarse en cada uno de los datos.

4.- ADMINISTRACION DE DATOS Y SUS HERRAMIENTAS DE SOFTWARE:

4.1.- Introducción:

A través de los capitulos anteriores, se han explicado las principales funciones de la administración de datos, así como sus principales estandares. Es evidente que estas actividades relacionadas con la administración de datos (planeación, diseño lógico, conversión logico-fisico, nomenclatura de datos, control del esquema de la base de datos, etc) son susceptibles de ser automatizadas por herramientas de software. **Es claro que la automatización de estas actividades constituye un factor de éxito de la administración de datos en la institución.**

Entre las actividades claves que deben automatizarse en la administración de datos, se encuentran: nomenclatura de datos y sus estándares, análisis y diseño logico de bases de datos, y control del esquema de la base de datos.

En este capítulo se muestran las caracteristicas de las herramientas de software que existen para automatizar a las principales actividades de la administración de datos. En estas caracteristicas, se mostrará que existen deficiencias para la automatización de la nomenclatura de datos y sus estándares, por lo que se hace necesaria la creación de una herramienta de software que permita crear, mantener, difundir y aplicar los estándares de nomenclatura de datos.

4.2.- Sistemas de diccionarios de datos:

Los diccionarios de datos (también conocidos como sistemas de diccionario/directorio de datos DD/DS) son herramientas usadas para recolectar, mantener y tener disponible la información existente sobre los datos de la institución.

Básicamente, un diccionario de datos provee mecanismos para definir y usar información sobre datos, campos, registros, archivos y bases de datos, así como las relaciones existentes entre estos elementos. También es capaz de manejar a otros elementos, tales como: reportes, pantallas, procesos, procedimientos, etc.

El diccionario de datos puede proveer información tal como:

- Nombre primario del dato.
- Definición del dato.
- Fecha de alta o última actualización.
- Nombres de programación usados bajo distintos ambientes.
- Relaciones existentes con otros datos.
- Valores válidos para cada dato.
- Formato físico de los datos.

A los diccionarios de datos también se les conoce como sistemas de diccionario/directorio de datos (DD/DS) en el

sentido de que no solo almacenan metadatos, sino que también proveen informacion de referencias cruzadas (directorios) sobre los metadatos, permiten tener la siguiente informacion:

- Relaciones existentes con:
  - Sistemas.
  - Programas.
  - Bases de datos.
  - Archivos.
  - Registros.

Asi, el diccionario de datos provee una vista lógica de los datos, mientras que el directorio de datos provee información de dónde residen fisicamente los datos (representados como campos de registros, variables de programas) y cómo se les puede tener acceso.

4.2.1.- Usos del diccionario de datos:

El diccionario de datos tiene los siguientes usos:

a) Como herramienta para compartir información.
b) Como glosario de definiciones.
c) Como herramienta de desarrollo y mantenimiento de sistemas.
d) Como herramienta para la estimación y análisis de impacto en cambios a los diseños lógicos de bases de datos.
e) Como un medio automatizado de documentación de datos.
f) Como un medio para la generación automatizada de las

descripciones físicas de los datos en diferentes medios de programación.

A continuación se explican cada uno de los usos del diccionario de datos:

a) Como herramienta para compartir información:

Al constituirse el diccionario de datos como un almacén centralizado de metadatos, se convierte entonces en un medio a través del cual, diferentes personas del departamento de sistemas puedan compartir información sobre los datos comunes a distintos sistemas y bases de datos.

b) Como glosario de definiciones:

Al almacenar las definiciones de los datos identificados en cada base de datos y proyectos del departamento de sistemas, es posible usar al diccionario de datos como un glosario. Este glosario permitiría obtener la información de la definición de cada dato y de esta forma estandarizar las definiciones de los datos para evitar datos redundantes debido a una pobre definición de los mismos.

c) Como herramienta de desarrollo y mantenimiento de sistemas:

El diccionario de datos es una herramienta efectiva para dar soporte al análisis y diseño

estructurado de sistemas. Para lo anterior, puede usarse para documentar el almacenamiento de datos, flujo de datos y procesos.

Con el diccionario de datos, es posible generar automáticamente la descripcion fisica de los datos que habrán de manejarse en los programas, asi como la estructura fisica de bases de datos, archivos, registros y campos. Al hacer lo anterior, se centraliza el control de las descripciones fisicas de los datos usados en los programas. Esto asegura la consistencia en el uso de los datos y reduce la redundancia de los mismos.

Como el diccionario de datos es la fuente de la descripción fisica de los datos, cualquier nuevo requerimiento debe hacerse bajo el conocimiento y aprobación de la administración de datos.

A lo largo del desarrollo de sistemas, el diccionario de datos estaría involucrado en las siguientes actividades:

1) Con la ayuda del grupo de administración de datos, los usuarios definen los datos que se usan en las funciones del negocio. Estos datos englobarian a la mayoria de los datos manejados por los sistemas de la institución. Estos datos se definirian y nombrarian de acuerdo a los estándares de nomenclatura de datos y se almacenarian

en el diccionario de datos.

2) Mediante la identificación de las características y relaciones de los datos almacenados en el diccionario de datos, se crea entonces un modelo lógico de base de datos.

3) Usando el modelo lógico de la base de datos, y considerando las características físicas del equipo de cómputo, DBMS (Sistema manejador de bases de datos) y sistema operativo, se diseña una base de datos física, en la que se consideran a las estructuras de datos más eficientes para el sistema. El diccionario de datos se encargaría de generar la descripción física de cada dato así como de generar los fuentes de las descripciones de las bases de datos y archivos involucrados en el desarrollo de sistemas.

d) Como herramienta para la estimación y análisis de impacto en cambios a los diseños de bases de datos:

Debido a que el diccionario de datos permite conocer con exactitud en dónde se están usando los datos, se le usa para analizar el impacto de un cambio en el diseño de bases de datos. De esta forma, si el formato físico de un dato cambia, se puede determinar en cuáles archivos habrá cambios, y cuales programas se verán afectados por este cambio. Esto es posible determinarlo

gracias a las referencias cruzadas que es capaz de generar el diccionario de datos.

e) **Como un medio automatizado de documentación de datos:**

Historicamente, los sistemas tradicionales de documentación han sido inadecuados debido a las siguientes razones:

- Muy poco tiempo se dedica a la documentación durante el desarrollo de un proyecto. Usualmente, la documentación se desarrolla después que se ha terminado de construir a los sistemas. Los esfuerzos por documentarlos son minimos, pues los diseñadores están ya preocupados por otros proyectos.

- La documentación no refleja en forma exacta cómo funciona realmente el sistema. Esto se debe a que la documentación se desarrolla después de haber construido al sistema. Con frecuencia esta documentación se encuentra incompleta. Aunque se instrumenten controles de documentación, no hay forma de garantizar que la documentacion reflejara el diseño y funcionamiento del sistema.

Para efectos de documentación, el diccionario de datos permite tener una documentación más confiable, debido a que el diccionario de datos

contiene la documentación completa sobre los datos. Al usar al diccionario de datos para generar los fuentes de la descripción física de los datos usados en los programas, garantiza la exactitud de la documentación. Después de que se ha implantado un sistema, todos los cambios que se hagan a los datos deberán hacerse primero en el diccionario de datos. Después el diccionario de datos se encargará de generar los fuentes con la descripción física de los datos de los programas. Esto asegura que la documentación se mantendrá al día y será más confiable.

f) Como un medio para la generación automatizada de las descripciones físicas de los datos en diferentes medios de programación:

Hasta ahora se ha mencionado que el diccionario de datos almacena a los datos y sus metadatos (nombre primario, nombres de programación, descripción lógica, fecha de actualización, tipo, longitud, etc). Con estos metadatos alimentados en el diccionario de datos, ya no es necesario escribir la descripción física de cada dato en los fuentes de los programas ni en las descripciones de las bases de datos físicas ni en la descripción de archivos. Mediante el diccionario de datos es posible la generación automática de los fuentes en donde se describe físicamente a los datos.

Cada cambio que se requiera hacer o nuevos datos que se quieran añadir al diseño de las bases de datos o archivos, ya no se hará directamente en los fuentes de los programas ni de las bases de datos físicas, sino que se hará en el diccionario de datos.

Posteriormente, el diccionario de datos generará la descripción física de los datos en los programas en los que se haya determinado que serían afectados por el cambio. Asimismo, el diccionario de datos se encargará de generar los fuentes actualizados de las bases de datos modificadas. De esta forma, en lugar de hacer cambios en "n" programas, se hace el cambio en el diccionario de datos y éste se encarga de generar los cambios en los "n" programas, archivos y bases de datos físicas.

Una grave deficiencia en los diccionarios de datos consiste en que no automatizan la nomenclatura de datos ni sus estándares. Los nombres primarios y nombres de programación que se almacenan en los diccionarios de datos, pueden estar estructurados en cualquier forma. No hay forma de validar si un nombre está estructurado correctamente, ni tampoco de verificar que se creó a partir de la definición del dato.

4.3.- Sistemas para control del uso de estándares de nomenclatura de datos:

Estos sistemas se utilizan como analizadores de programas fuentes. El análisis que realizan estos sistemas consiste en revisar la declaración de:

- Campos de archivos
- Campos de registros
- Variables

Estos sistemas revisan que estas declaraciones estén hechas de acuerdo a los estándares de nomenclatura de datos. De esta forma, en estos sistemas se indican las reglas de abreviación adoptadas por ambiente de programación, así como de las palabras clase y palabras entidad (y sus correspondientes abreviaturas convenidas) consideradas como válidas por la administración de datos.

Al realizar el análisis de los programas fuentes, estos sistemas analizadores son capaces de detectar declaraciones de variables que no corresponden a los estándares, indicando el tipo de error y la línea de programa en donde detectaron dicho error.

La principal deficiencia de estos sistemas, consiste en que sólo se usan una vez que los programas a analizar ya se encuentran codificados. Es decir, no ayudan a aplicar los estándares de nomenclatura de datos, sólo a revisar su aplicación. Con este tipo de herramientas, no es posible generar los nombres primarios y nombres de programación a partir de la definición de los datos.

4.4.- Sistema de nomenclatura de datos:

Como se ha explicado en las secciones anteriores, las herramientas hasta ahora disponibles para la administración de datos, no son capaces de automatizar la nomenclatura de datos ni sus estándares.

El principal problema que existe para que los estándares de nomenclatura de datos sean adoptados con éxito en el área de sistemas de cualquier institución, consiste en la aplicación correcta de las reglas de abreviación, así como la actualización de las listas de palabras clase, palabras entidad (y sus abreviaturas convenidas) consideradas como válidas. Este problema es aún mayor, si se considera que las reglas de abreviación pueden ser distintas entre cada ambiente de programación y diccionarios de datos que existan en la institución. Si no se aplican correctamente los estándares de nomenclatura de datos, entonces la generación de los nombres primarios y nombres de programación de los datos corre el riesgo de caer en el fracaso:

Podrían generarse nombres incorrectos a partir de definiciones incompletas.

Podrían generarse nombres en los que las abreviaturas se construyen sin aplicar correctamente las reglas de abreviación. Lo anterior daría como resultado la proliferación de

distintos nombres para un mismo dato.

Cada área del negocio podría tener listas distintas e incompletas de palabras clase, palabras entidad y abreviaturas convenidas. Lo anterior daría como resultado que los datos fuesen difícilmente compartidos entre diferentes áreas del negocio y sistemas, pues los nombres de los datos serían distintos y por tanto sería difícil identificarlos como comunes.

Ante esta situación, es evidente que es necesario desarrollar una herramienta de software que sea capaz de automatizar la nomenclatura de datos y sus estándares.

Como parte de esta tesis, se desarrolló un sistema capaz de crear, mantener y aplicar los estándares de nomenclatura de datos. En el capítulo 5 se describen sus principales características, funciones y productos, así como los fundamentos teóricos que se estudiaron para darle origen. A dicha herramienta se le ha denominado **Sistema de Nomenclatura de Datos.**

Ahora bien, debe considerarse que esta herramienta de software deberá estar relacionada con las otras herramientas que se analizaron en el presente capítulo. A continuación se explica esta relación:

El administrador de datos por área de negocio, o por

proyecto, estara encargado de especificar la definición del dato que se desea nombrar, de forma tal que el sistema de nomenclatura de datos generará el nombre primario y los nombres de programación. El nombre primario y la definicion del dato serán entregados al experto en el manejo del DD/DS, para que el diccionario de datos se actualice con la nueva información. Asimismo, los nombres de programación serán registrados en el diccionario de datos, indicando su relación con programas que hacen uso de ellos, así como los archivos y bases de datos en los que forman parte como campos.

Los nombres de programación serán distribuidos a los programadore que requieran de ellos durante el desarrollo de los sistemas, de forma tal que los programadores no requieran memorizar listas de palabras clase, palabras entidad ni abreviaturas especiales. Tampoco necesitarán memorizar reglas de abreviación. La generación de los nombres se hará con el sistema, a partir de la definición de los datos que habrán de usarse en los sistemas en desarrollo.

Una vez que los programadores y personal involucrado en el desarrollo de sistemas han codificado los programas, los fuentes de éstos serán analizados por los sistemas usados para controlar el uso de estándares. El proposito de esta acción consiste en validar que realmente se usaron los nombres generados por el sistema de nomenclatura de datos. El administrador de datos corporativo será la única persona que tenga la autoridad para añadir o actualizar reglas de

abreviación en el sistema de nomenclatura de datos. Asimismo, podrá actualizar las listas de palabras clase, palabras entidad y abreviaturas convenidas, y tendrá la responsabilidad de distribuirlas a los usuarios del sistema.

El diagrama 4.4.1 muestra la relación del sistema de nomenclatura de datos con sus usuarios y otras herramientas de la administración de datos.

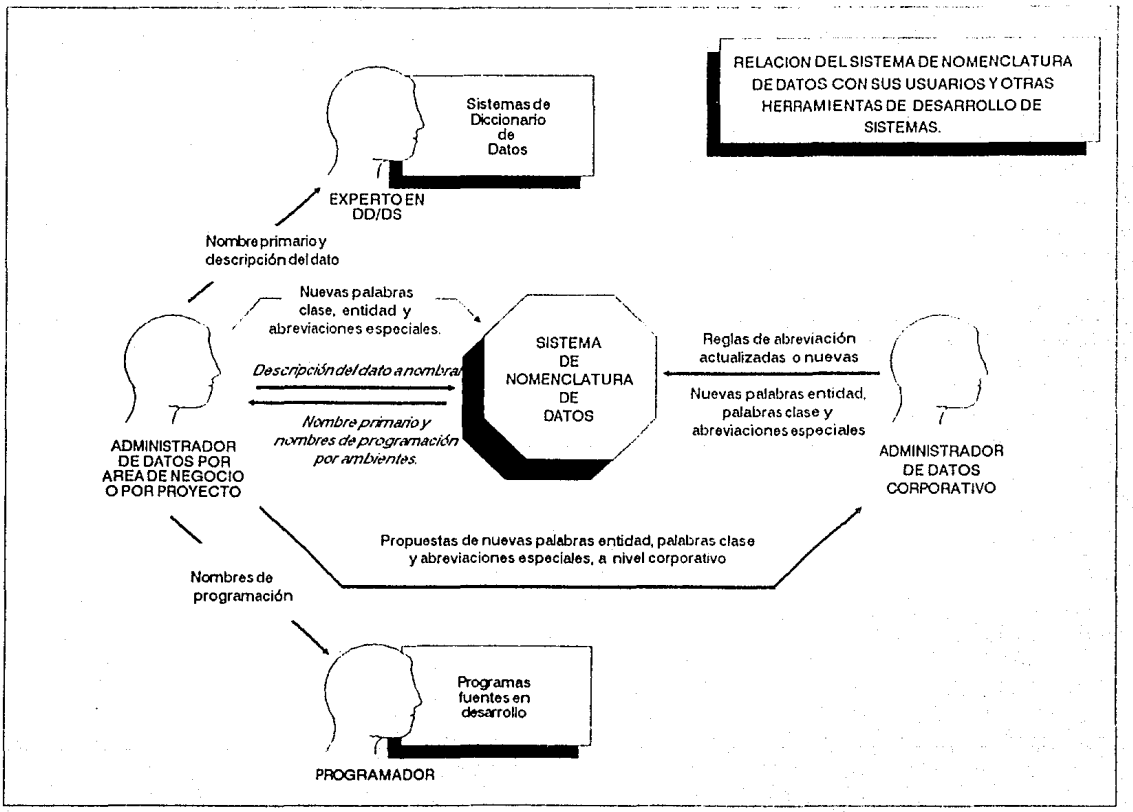


Diagrama 4.4.1

4.5.- Red semántica para presentar la relación existente entre la administración de datos, nomenclatura de datos y herramientas afines.

En los capítulos anteriores se han analizado los puntos referentes a la adminstración de datos: funciones, beneficios, actividades y herramientas que le brindan soporte.

Ahora bien, en esta sección se presenta una forma de representar al conocimiento expresado a lo largo de estas hojas. Su propósito consiste en brindar al lector una vista panorámica que le permita reconocer las relaciones entre la administración de datos, nomenclatura de datos y sus herramientas.

Para representar el conocimiento, existen diversas técnicas para este fin; entre ellas, se encuentra la técnica de las **redes semánticas**.

Una red semántica es un conjunto de **objetos**, éstos pueden ser:

- Objetos físicos que pueden verse o tocarse.
- Entidades conceptuales tales como actos, eventos o categorías abstractas.

Los objetos están a su vez relacionados entre sí a través de **asociaciones**. Las asociaciones más comunes son las siguientes:

- Asociaciones ejemplificativas:

A través de estas asociaciones, se indica que una serie

de objetos son ejemplos u ocurrencias de otro objeto.

- Asociaciones de propiedad:

A través de estas asociaciones, se indica que un objeto representa una propiedad inherente de otro objeto.

- Asociaciones definicionales:

A través de estas asociaciones, se indican todo tipo de relaciones que existan entre 2 objetos de la red semántica.

Las redes semánticas poseen la propiedad de **herencia.** A través de esta propiedad, un objeto **"hereda"** las características de otro objeto con el que está relacionado. La propiedad de herencia es una implicación de las asociaciones ejemplificativas. Esta propiedad indica que que las ocurrencias o ejemplos de un objeto tienen las mismas características o propiedades del objeto al cual ejemplifican.

En la figura 4.5.1 se muestra la red semántica usada para representar las relaciones existentes entre la administración de datos, nomenclatura de datos y herramientas afines:

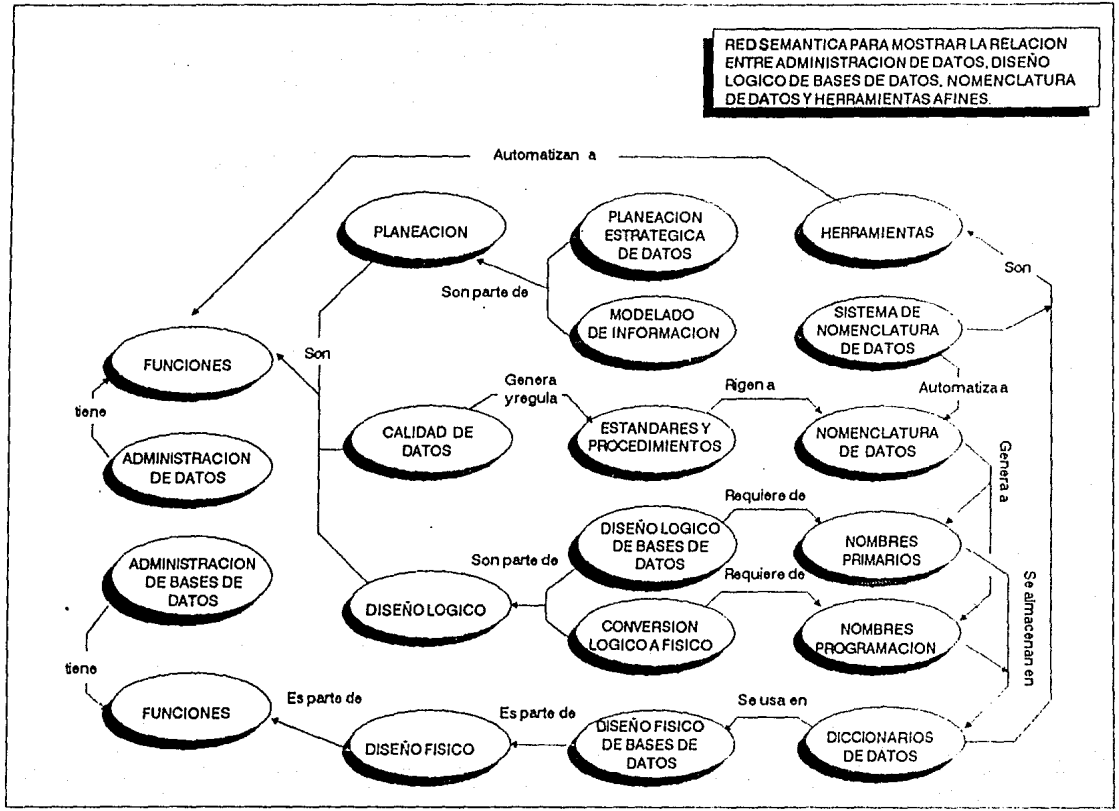


Diagrama 4.5.1

5.- SISTEMA DE NOMENCLATURA DE DATOS:

5.1.- Introducción:

En la seccion 4.4. se indicó la necesidad de una herramienta de software que automatice la nomenclatura de datos y sus estándares. Como parte de la presente tesis, se desarrolló un sistema encargado de esta automatización, y al cual se le ha denominado **Sistema de Nomenclatura de Datos.** En este capitulo se presenta su descripción funcional, en la que se muestran las premisas que se consideraron para el diseño y construcción del sistema. Se consideró fundamental el uso de la teoría de lenguajes formales y autómatas por 2 razones:

1) El sistema está orientado a un ambiente de ventanas. El manejo de ventanas y teclado es mucho más fácil de hacerse a través de autómatas que controlen la secuencia de teclas que manejan las ventanas del sistema.

2) Es evidente que el sistema debe ser capaz de analizar la definición del dato y determinar si ésta se encuentra correctamente estructurada y conteniendo palabras claves que permitan la generación del nombre primario y nombres de programación. Por ello, se consideraron las características de un lenguaje desarrollado por I.B.M. para la nomenclatura de datos (llamado **lenguaje OF**) y se adecuaron a las características del lenguaje español, dando como resultado al **lenguaje de nomenclatura de datos.**

Asimismo, se presenta un análisis del lenguaje español, para que de esta forma se considerasen sus principales características al adecuar al lenguaje OF y dar como resultado al lenguaje de nomenclatura de datos. Por otra parte, se consideró que el sistema debería tener la inteligencia suficiente para poder determinar si una palabra corresponde realmente al lenguaje español. Para ello, se consideró que debería existir alguna característica entre las palabras del lenguaje español. Por ello, se presenta también un análisis morfológico de los sustantivos y adjetivos del lenguaje español. En base a dicho análisis, el sistema es capaz de determinar si una secuencia de letras es realmente una palabra del lenguaje español, sin la necesidad de tener un gran diccionario de palabras.

Posteriormente, se muestra el diseño lógico del sistema, explicando cada uno de los autómatas encargados del control del lenguaje de nomenclatura de datos, manejo de errores, y manejo de ventanas.

Más adelante, se muestra el diseño físico, en donde se explican las consideraciones que se siguieron para el manejo del ambiente orientado a ventanas, estructuras de datos usadas para manejar las palabras válidad del sistema, manejo de archivos y el manejo de procesos de validación y seguridad del sistema.

Por último, se hace una descripción de las principales rutinas del sistema de nomenclatura de datos.

5.2.- Descripción funcional:

Una de las funciones de la administracion de datos consiste en dictar estándares de nomenclatura de datos que rijan la construcción de los nombres de los datos de una institución.

Sin embargo, las reglas de abreviación no pueden aplicarse el 100% de las ocasiones, sino que pueden existir datos cuya abreviación no corresponde a las reglas dictadas por la administración de datos, pero que por su frecuencia de uso son aceptadas en toda la institucion.

No sólo debe considerarse la definición y mantenimiento de reglas de abreviación, sino que también debe considerarse que se requiere que el sistema ayude a estructurar correctamente la definición del dato. Para ello, se desarrolló al lenguaje de nomenclatura de datos, el cual indica la secuencia correcta de las palabras claves que dan la información necesaria para que la definición del dato sea completa y que a partir de ella se generen el nombre primario y los nombres de programación de los datos.

Ahora bien, si se desea pensar en una herramienta de software que dé soporte a estas funciones, deben considerarse los siguientes requerimientos iniciales:

5.2.1.- Requerimientos iniciales :

1) El sistema debe permitir al administrador de datos dictar las reglas usadas para la construcción de abreviaciones en la nomenclatura de datos.

2) El sistema debe permitir al administrador de datos cambiar estas reglas de construcción de abreviaciones, y actualizar en forma automática su acervo de abreviaturas.

3) El sistema debe permitir al administrador de datos incluir palabras cuya abreviatura no obedece a las reglas dictadas por la administración de datos.

4) El sistema debe permitir al administrador de datos especificar la definición de un dato, para que de esta forma el sistema genere el nombre del dato.

5) El sistema, al recibir el texto de definición de datos, debe ser capaz de analizarlo y hallar las palabras claves que permitan generar el nombre del dato, o bien, que indique que el texto no es lo suficientemente explicativo.

6) Al analizar el texto de definición, el sistema debe ser capaz de determinar si existen nuevas palabras claves, de forma tal que actualice su acervo de palabras claves y sus correspondientes abreviaturas.

7) Este análisis debe contemplar el hecho de que una palabra mal escrita debe ser detectada por el sistema, antes de actualizar su acervo de palabras claves.

8) Este análisis debe contemplar el hecho de que una palabra puede ser el sinónimo de una palabra clave existente en el acervo del sistema, por lo que el sistema debe ser capaz de detectar sinónimos antes de actualizar su acervo de palabras claves.

Hasta este punto, es claro que el sistema debe regirse por un autómata que siga la sintaxis del lenguaje de nomenclatura de datos, de forma tal que le permita validar (aunque no en forma completa) la definición del dato que se quiere nombrar. También es factible pensar la presencia de diccionarios, los cuales se encarguen de las funciones de revisiones ortográficas así como la revisión de sinónimos. También se requiere la existencia de un acervo de palabras claves con sus correspondientes abreviaturas (para el caso en el que las abreviaturas no cumplan las reglas de construcción dictadas por la administración de datos).

9) El sistema debe considerar el significado de la definición del dato a nombrar para que, de esta forma, la generación de los nombres sea correcta. Por ello, se requiere que se haga un análisis semántico de la definición a partir del uso de lexemas y gramemas así

como de ciertas reglas de inteligencia.

10) Debe ser capaz de establecer una serie de diálogos que permitan esclarecer una definición confusa o incompleta.

11) Estos diálogos deben generar una serie de acciones, las cuales pueden ir desde el rechazo del texto de definición, su modificación o la actualización del acervo de palabras clase, entidad y reglas de abreviación.

5.2.2.- Objetivo principal del sistema:

El objetivo principal del Sistema de Nomenclatura de Datos es:

Proporcionar apoyo automatizado en la definición, mantenimiento y aplicación de estándares y procedimientos para la nomenclatura de datos.

5.2.3.- Funciones del sistema:

Las funciones del Sistema de Nomenclatura de Datos son las siguientes:

1) **Apoyar al administrador de datos en la aplicación del lenguaje de nomenclatura de datos.**

   Para lograr lo anterior, se desarrolló un lenguaje denominado **lenguaje de nomenclatura de datos**, el cual se basa en los conceptos del **of language** y

considera características propias del lenguaje español.

El sistema ayuda al administrador de datos en la aplicación del lenguaje de nomenclatura de datos mediante la presencia de una serie de mensajes, en los que el sistema indica el elemento del lenguaje que el usuario debe escribir para formar la definición del dato a nombrar.

El sistema es capaz de hallar la semejanza existente entre palabras, de forma tal que impide que se escriban versiones incorrectas de palabras clase y entidad. Por ejemplo: Al escribir la palabra "clientela" el sistema indicará que existe una palabra entidad "cliente" que es la correcta; el sistema permitirá al usuario escribir otra palabra o cambiar automáticamente "clientela" por "cliente" en el texto de definición del dato.

El sistema también se encarga de la detección de errores al momento de especificar la definición del dato a nombrar. Esta detección del error debe indicarse mediante una serie de diálogos con el usuario, de forma tal que se le explica el tipo de error y la forma en que puede darle solución.

El sistema debe permitir al usuario corregir, en cualquier momento, la definición que está especifi-

cando. De esta forma, puede borrar parte de la misma y continuar especificándola más tarde.

2) Proporcionar al administrador de datos la capacidad de especificar reglas de abreviación para diferentes diccionarios de datos y ambientes de programación.

De esta forma, el administrador de datos puede tener diferentes reglas de abreviación para distintos diccionarios de datos y ambientes de programación.

3) Permitir al administrador de datos actualizar el conjunto de palabras clase y entidad, consideradas como válidas por la administración de datos.

Al intentar actualizar al acervo con una nueva palabra, el sistema revisará si la palabra, se encuentra en singular o plural. En caso de encontrarse en plural, genera la versión en singular de la palabra y lo indica al administrador de datos. De esta forma, sólo se almacenarán las palabras en singular, pero con la versatilidad de permitir el uso del plural en la definición de los datos. El sistema se encarga de generar el singular de las palabras, en base al estudio realizado en los gramemas característicos del singular de las palabras en el

lenguaje español.

El sistema también revisa que una palabra que se quiere dar de alta como palabra clase no sea una palabra entidad, preposición, articulo u otro elemento del lenguaje de nomenclatura de datos.

También se revisa que una palabra que se quiere dar de alta como palabra entidad no sea una palabra clase, preposición, articulo u otro elemento del lenguaje de nomenclatura de datos.

El sistema permite al usuario indicar que cada palabra clase o entidad tendrán una abreviatura especial o si se construirá a partir de las reglas de abreviación. En caso de especificar abreviatura especial, el sistema investiga si la nueva abreviatura es igual a cualquier otra abreviatura existente en el acervo. En caso afirmativo, indica al usuario que deberá especificar otra abreviatura diferente a la anterior.

4) Una vez que el usuario ha especificado la definición del dato a nombrar, el sistema genera el nombre primario y los nombre de programación.

Para ello, el usuario especifica el nombre del

diccionario de datos o el nombre del ambiente de programación para el que desea generar el nombre. La elección es interactiva, así que una vez generado el nombre primario o de programación, el usuario puede indicar otro diccionario de datos o ambiente de programación.

El sistema muestra al usuario las acciones que realiza para abreviar cada palabra clave de la definición del dato para formar las abreviaturas que forman parte del nombre del mismo.

En cualquier momento, el administrador de datos puede consultar las reglas de abreviación utilizadas en el momento de la generación de los nombres del dato.

5) Control de los usuarios del sistema:

De acuerdo a la organización de administradores de datos de la institución, existirán diferentes categorías de estos. De acuerdo a la categoría de cada administrador de datos, será función del sistema llevar el control de las diferentes operaciones que le estarán permitidas a cada categoría.

Existen 3 categorías de administradores de datos:

- Administrador de datos corporativo.

- Administrador de datos por área del negocio.
- Administrador de datos por proyecto.

El administrador de datos corporativo podrá realizar las siguientes funciones con el sistema:

- Dar de alta o actualizar palabras clase.
- Dar de alta o actualizar palabras entidad.
- Dar de alta o actualizar nombres de sistemas de diccionario de datos para los cuales se generarán reglas de abreviación.
- Dar de alta o actualizar nombres de ambientes de programación para los cuales se generarán reglas de abreviación.
- Actualizar reglas de abreviación para sistemas de diccionario de datos o ambientes de programación especificados en el acervo del sistema.

Los administradores de datos por área del negocio, podrán realizar las siguientes funciones:

- Generar los nombres primarios de los datos, a partir de su definición. Estos nombres primarios posteriormente se almacenarán en cada diccionario de datos para los que fueron generados.
- Generar los nombres de programación a partir de la definición del dato. Estos nombres de progra-

mación deberán darse de alta en los diccionarios de datos como nombres de campos o variables que representan a al dato nombrado.

- Dar de alta o actualizar palabras clase, entidad, así como sus respectivas abreviaturas.

Los administradores de datos por proyecto podrán:

Generar los nombres primarios y de programación de los datos que desean nombrar.

5.2.4.- Productos del sistema:

Los productos del Sistema de Nomenclatura de Datos, son los siguientes:

1) Nombre primario:

El sistema generará el nombre primario del dato para cada diccionario de datos cuyas reglas de abreviación hayan sido especificadas al sistema.

2) Nombre de programación:

El sistema generará el nombre de programación del dato para cada ambiente de programación cuyas reglas de abreviación existan en el acervo del sistema.

3) Acervo de palabras clase y entidad, con sus correspondientes abreviaturas convenidas.

El sistema lleva el control del acervo de

palabras clase y entidad, así como de sus abreviaturas. Este acervo es histórico, por lo que se contempla que en futuras versiones del sistema podrá consultarse la evolución de las abreviaturas convenidas de cada palabra, así como del acervo.

Este mismo acervo puede ser compartido entre distintos administradores de datos por área de negocio, por lo que puede analizarse cuáles palabras clase y entidad son comunes a todas las áreas del negocio.

4) Acervo de reglas de abreviación por cada diccionario de datos y ambientes de programación indicados por la administración de datos.

Este acervo de reglas de abreviación también será compartido por los distintos administradores de datos, por lo que será responsabilidad del administrador de datos corporativo indicar la periodicidad con la que se actualizará el acervo entre los distintos administradores de datos de las áreas del negocio y por proyecto de la institución, así como verificar que se lleve correctamente lo anterior. Esto es con el propósito de tener las mismas reglas de abreviación aplicándose en todos los proyectos de sistemas.

Este acervo también es histórico, por lo que en versiones futuras del sistema existirá la posibilidad de analizar la evolución de las reglas de abreviación para cada diccionario de datos y ambiente de programación.

5) **Generación de reportes del Sistema de Nomenclatura de Datos:**

El sistema de nomenclatura de datos es capaz de generar documentos en forma de archivos magnéticos, almacenados en el disco duro, y que son reconocibles por **GEM 1st Word Plus.** De esta forma, los reportes generados pueden ser analizados por los administradores de datos y susceptibles de modificaciones antes de mandarse a impresión. Los administradores de datos, a través de las facilidades del Sistema de Nomenclatura de Datos, pueden definir el formato de cada reporte de acuerdo a los siguientes reportes estándares:

1.- Reporte de palabras clases y abreviaciones.

2.- Reporte de palabras entidad y abreviaciones.

3.- Reporte de diccionarios de datos.

4.- Reporte de ambientes de programación.

5.- Reporte de reglas de abreviación que pertenecen a los diferentes diccionarios de datos y ambientes de programación.

6.- Reporte sobre abreviaturas convenidas.

7.- Reporte sobre los elementos eliminados:
   - Palabras clase.
   - Palabras entidad.
   - Diccionarios de datos.
   - Ambientes de programación.

8.- Reporte sobre los nombres primarios y nombres de programación generados durante la sesión.

El administrador de datos tiene la facilidad de estructurar su documento, de forma tal que incluya a todos o sólo a algunos de los reportes anteriores, en el orden que desee. No existe restricción en cuanto a la longitud que pueda tener el archivo generado, excepto el espacio disponible en disco.

5.3.- Fundamentos teóricos:

5.3.1.- Introducción:

El desarrollo del sistema de nomenclatura de datos, implicó no sólo el desarrollo de un lenguaje que permitiese aplicar los conceptos de nomenclatura de datos, sino que también permitió aplicar la teoría de lenguajes formales y autómatas al manejo de ventanas y teclado del sistema. En la siguiente sección se muestran los conceptos básicos de lenguajes formales y autómatas. Más adelante, se indicarán las características del **lenguaje OF**. En esta sección será evidente la necesidad de adecuar las características de este lenguaje a las necesidades idiomáticas de los usuarios que tendrán el acceso al sistema. Debido a que el sistema de nomenclatura de datos fue desarrollado para usuarios de habla española, fue necesario revisar las características del lenguaje español. Una vez realizada esta revisión y tomando en cuenta la filosofía del **lenguaje OF**, se desarrolló (en base a la teoría de lenguajes formales y autómatas) al **lenguaje de nomenclatura de datos.**

Deseando que el sistema de nomenclatura de datos presentara la mayor versatilidad posible, y que a su vez brindara el mejor apoyo inteligente al usuario, se determinó que el sistema debería lograr identificar el plural y el singular de cada palabra, así como cambiar una palabra escrita en plural a singular y viceversa. Asimismo, se determinó que debería este análisis basarse en la presencia de los gramemas y lexemas que forman parte de las palabras

del lenguaje español. Posteriormente se vio que ciertas terminaciones de las palabras (gramemas) son característicos de sustantivos, y otros de adjetivos. Esto se mostró muy atractivo para el análisis sintáctico y semántica que debería realizar el sistema de nomenclatura de datos para validar la definición de los datos que habrían de nombrarse. Por esta razón, en esta tesis existe una sección que muestra la ventaja del uso de gramemas en el sistema de nomenclatura de datos.

5.3.2.- Introducción a lenguajes formales y autómatas:

- **Lenguajes:**

Un **alfabeto** es un conjunto de **símbolos** que se usan para formar palabras, estos símbolos también reciben el nombre de **símbolos terminales.** Por citar un ejemplo, las letras forman los símbolos terminales del alfabeto del lenguaje español.

Una **palabra** es la concatenación de los símbolos de un alfabeto, y se le considera como una unidad. Una **cadena** u **oración** es la concatenación de palabras.

Un **lenguaje** es el conjunto de todas las posibles oraciones que pueden formarse a partir del alfabeto. Es evidente que muchas de estas cadenas u oraciones no tendrían significado, o ni siquiera estar construidas en forma razonable.

Así un lenguaje debe considerar no sólo a las cadenas que puedan formarse a partir del alfabeto, sino que deben escogerse a aquellas que estén estructuradas correctamente y que además tengan algún significado.

A la especificación de la construcción correcta de cadenas u oraciones se le conoce como **sintaxis.** A la especificación del significado de las oraciones se le denomina **semántica.**

Considerando lo anterior, un lenguaje puede representarse formalmente a través de ciertas reglas y símbolos que

definen la sintaxis del lenguaje. A este conjunto de reglas y símbolos se le denomina **gramática**.

Formalmente, una **gramática** se compone por 4 elementos:

$$G = \{ N , T , P , S \}$$

donde: **N** es el conjunto de símbolos no terminales, también llamados "categorías sintácticas".

T es el conjunto de símbolos terminales sobre los cuales está definiéndose el lenguaje. Dicho conjunto forma el alfabeto del lenguaje.

P es el conjunto de reglas de producción, las que indican la forma en que pueden relacionarse los símbolos terminales y no terminales para producir cadenas del lenguaje.

S es un elemento de N con el cual debe iniciarse cualquier generación de cadenas.

Los **símbolos terminales** forman parte del alfabeto. Los **símbolos no terminales** corresponden a **categorías o funciones sintácticas** que pueden tener las palabras en un lenguaje. Así, en el lenguaje español pueden ejemplificarse varias categorías sintácticas: sujeto, complemento, verbo, núcleo nominal, sustantivo, adjetivo, adverbio, etc. Algunas palabras que corresponden a estas categorías sintacticas son: verbo (jugar), sustantivo (niño), adjetivo (preferido), adverbio (diariamente), etc. De estos ejemplos, hay algunas categorías sintácticas que pueden formarse a partir de otras, o bien formarse por palabras. La forma en que se representa estas relaciones existentes entre diferentes elementos no

terminales (categorías sintácticas) y los **elementos terminales** del lenguaje (palabras o símbolos del alfabeto) es mediante las **reglas de producción.**

Existen varias notaciones para representar a las reglas de producción. En la presente tesis se usa la notación **Backus-Naur (BNF)**, la cual se explica a continuación:

En esta notación, los símbolos no terminales o categorías sintácticas se escriben entre llaves < >. Así por ejemplo, las categorías sintácticas mencionadas en los párrafos anteriores se escribirían así: <sujeto>, <predicado>, <verbo>, <núcleo nominal>, <sustantivo>, <adjetivo> , etc.

Cada regla de producción se compone por 2 miembros: miembro izquierdo y miembro derecho, los cuales están separados por el símbolo ::= que se lee como **"se define como"** o **"produce a".**

Tanto el miembro izquierdo como el miembro derecho de la regla de producción están formados por símbolos no terminales y terminales. Dependiendo del tipo y cantidad de los símbolos que se encuentran en cada miembro, las gramáticas se clasifican en varios tipos (los que se explican más adelante).

Esta estructura de las reglas de producción (miembros izquierdo y derecho) permite indicar que el miembro izquierdo de la regla de producción podrá ser sustituido por el miembro derecho, siempre que el miembro izquierdo de la

misma aparezca en una oracion. A la operacion de tomar una regla de producción para sustituir a los elementos del miembro izquierdo por los elementos del miembro derecho en una oración, se le llama **derivación.**

Ahora bien, en esta notación se tiene contemplado el caso en el cual un miembro derecho de la regla de producción pudiera estar definido en términos de varios miembros derechos. En este caso, se usa al símbolo | separando a los diferentes miembros derechos y sirve para indicar que puede escogerse a cualquiera de esos miembros derechos de la regla de producción.

Asi, siguiendo esta notación, si se quiere indicar que un núcleo nominal está formado por un sustantivo o bien por un sustantivo y un adjetivo, se escribiría la siguiente regla de producción:

<núcleo nominal> ::= <sustantivo> | <sustantivo><adjetivo>
O pudo haberse escrito asi:
<núcleo nominal> ::= <sustantivo>
| <sustantivo><adjetivo>

Puede darse el caso en el que en el miembro izquierdo de una producción aparezca también entre los símbolos que forman al miembro derecho. En tal caso, se dice que la regla de producción es **recursiva.** Si una regla de producción recursiva tiene un simbolo no terminal en el miembro izquierdo, se dice que la regla de producción es **normal,** si dicho elemento

no terminal aparece en el miembro derecho una sola vez y es el símbolo que se encuentra en el extremo más a la derecha. Por ejemplo, para representar a los números enteros en los lenguajes de programación, se usan las siguientes reglas de producción: (una de ellas es recursiva y normal)

<número entero> ::= <digito>
| <digito><número entero>
<digito> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Anteriormente se mencionó que dependiendo del tipo y la cantidad de los símbolos que forman parte del miembro derecho de las reglas de producción, es como se clasifican las gramáticas. Existen los siguientes tipos de gramáticas:

Tipo 0 : No existe restricción alguna en las reglas de producción.

Tipo 1 : En cualquiera de las reglas de producción, la cantidad de símbolos que forman parte del miembro derecho es mayor a la cantidad de símbolos que forman parte del miembro izquierdo.

Tipo 2 : El miembro izquierdo de cada regla de producción está formado por un sólo símbolo no terminal y el miembro derecho consta de uno o varios símbolos (terminales o no terminales).

Tipo 3 : El miembro izquierdo de cada regla de producción está formado por un sólo símbolo no terminal y el miembro derecho consta de uno o varios símbolos incluyendo a lo sumo un símbolo no terminal, el cual debe estar colocado en el extremo derecho.

Cada tipo de gramática es un caso particular del tipo precedente. Así, las gramáticas del tipo 3 son un caso del

tipo 2. Las de tipo 2 son un caso del tipo 1 y las de tipo 1 son un caso de las de tipo 0.

Las gramáticas de tipo 1 se les conoce como **gramáticas sensibles al contexto**, pues la derivación que se haga con algún elemento no terminal, dependerá de la posición y elementos que se encuentran junto a él. Las gramáticas de tipo 2 se les conoce como **gramáticas libres de contexto**, pues los símbolos del miembro izquierdo de las reglas de producción se sustituyen en donde quiera que aparezcan, sin importar los símbolos que los rodean.

A las gramáticas del tipo 3 se les conoce como **gramáticas regulares.**

A un lenguaje se le llama de **tipo 2 (libre de contexto)** o de **tipo 3 (regular)** si existe una gramática de estos tipos que lo originen.

- Autómatas

Un autómata finito "A" está formado por 5 elementos:

$$A = (K,T,M,q_0,H)$$

Donde: **K es un conjunto finito y no vacío de estados.**

T es un alfabeto o conjunto de símbolos.

M es una función que mapea del conjunto K X T a K.

q_0 **K es el estado inicial del autómata.**

H es un subconjunto de K y representa a los estados de fin de ejecución del autómata.

La forma en que trabaja un autómata finito, es la siguiente:

La entrada del autómata está formada por una secuencia de símbolos que forman parte del conjunto **T**. El autómata se encarga de leer uno por uno a estos símbolos, y dependiendo del estado actual del autómata y del simbolo de entrada leído, se hace una transición a otro estado. Esta transición está regida por la función de transición de estados. Esta función está determinada por una matriz de transición de estados, en la que los renglones representan a los estados **K** del autómata, y las columnas representan a los símbolos de entrada **T**. La función de transición de estados podría expresarse de la siguiente manera:

$$k_{a+1} = M [k_a, t_a]$$

En donde: k_{a+1} representa al siguiente estado del autómata, el cual es el resultado de la función de transición de estados.

M representa a la matriz de transición de estados, la cual indica cómo habrán de efectuarse las transiciones, dado el estado actual y el símbolo de entrada que se lea en dicho estado.

k_a representa al estado actual del autómata.

t_a representa al símbolo leído en el estado actual del autómata.

En cada estado, el autómata leerá un símbolo de entrada, efectuará una acción determinada y posteriormente efectuará la transición a otro estado. Este proceso se repetirá hasta que el autómata finito alcance alguno de los estados **H** que

corresponden al fin de ejecución, o bien que ya no existan más símbolos T que leer.

Se dice que un automata finito es **determinístico** cuando se cumple que la función de transición de estados genera un sólo resultado o transición, dado el estado actual y un símbolo de entrada. Un autómata finito es **no determinístico** cuando la función de transición de estados genera transiciones a más de un estado del autómata.

5.3.3.- Lenguaje OF:

El lenguaje OF para la nomenclatura de datos se desarrolló en I.B.M. con el propósito de estandarizar los nombres de los datos que habrían de almacenarse en los diccionarios de datos. Su propósito principal consiste en crear una plataforma en la que diferentes personas, trabajando en forma independiente, pudieran generar el mismo nombre para el mismo dato. Al alcanzarse este objetivo en la institución, se eliminaría el problema de tener diferentes nombres para un mismo dato, así como nombres iguales para diferentes datos. Al mismo tiempo, el control del diccionario de datos se vería simplificado al manejar sólo nombres estándares.

Hay que considerar que este lenguaje fue desarrollado en base al lenguaje inglés, por lo que a continuación se muestran algunas características que corresponden al lenguaje inglés.

Al tratar de nombrar un dato, debe hacerse a traves de definición, la que debe contener información mínima y suficiente para lograr nombrarlo. Esta información debe contestar a las preguntas: ¿Qué es? y ¿De quién es?, así como cierta información complementaria que permita diferenciar al dato de otros semejantes.

Recuérdese que en el idioma inglés los adjetivos modifican a los sustantivos en orden inverso al lenguaje español: primero se presenta el adjetivo y después el

sustantivo. P.ej.: "Data naming conventions" que en español sería "Convenciones para nomenclatura de datos".

Debido a la característica anterior, se pensó que sería muy difícil identificar cuáles palabras brindarían la información mínima y suficiente para nombrar a los datos. Por esta razón, se pensó en modificar el orden de las palabras en la definición del dato, de forma tal que el orden se viese regido por el uso de preposiciones. De esta forma, el ejemplo anterior se vería modificado a: "Conventions for data naming". El uso de las preposiciones, (o conectores como los llaman en los documentos originales) llevó a que el orden de las palabras fuese más parecido al orden que se lleva en el lenguaje español. Ahora bien, se vió que de las preposiciones utilizadas, la gran mayoría correspondía a la preposición OF por lo que por esta razón se decidió nombrar a dicho lenguaje como **LENGUAJE OF.**

El lenguaje OF clasifica a las palabras en 3 categorías:

- Class words (palabras clase), que dan respuesta a la pregunta ¿Qué es?
- Prime words (palabras entidad), que dan respuesta a la pregunta ¿De quién?
- Modifiers (modificadores), que dan información que permite diferenciar datos semejantes.

El orden de las palabras en la definición del dato, de acuerdo al lenguaje OF, es de tipo jerárquico. Es decir, el primer elemento o palabra es la que brinda la información más general, la siguiente un poco menos general, hasta que la

última viene a ser la que brinda la información más especí-fica.

Al tratar de aplicar el lenguaje OF se indicó, en base al uso de preposiciones, el siguiente orden de palabras en la definición del dato a nombrar:

<palabra_clase><conector><palabra_entidad><1er modificador> <conector><2o. modificador> .. <conector><último modificador>

Los conectores podrían ser preposiciones o palabras o frases como: **y, que es, que son, o.** y que no se usarían al momento de generar el nombre primario o de programación del dato.

A continuación se muestran algunos ejemplos de cómo se aplica el lenguaje **OF** a partir de la definición del dato en lenguaje inglés:

Definición del dato en idioma inglés: **work unit code**
Definición del dato en lenguaje OF : **code of unit of work**
Palabra clase : **code**
Palabra entidad: **unit**
Modificador : **work**
Nombre primario sin abreviaturas : **code-unit-work**
Definición del dato en idioma inglés: **employee's hire date**
Definición del dato en lenguaje OF : **date of hire of employee.**

Como puede verse en este último ejemplo, es difícil clasificar las palabras en base al orden que guardan en la

definición del dato. Sin embargo, se encuentra cierta similitud entre el lenguaje OF y el lenguaje español. Por lo anterior, se determinó hacer una revisión del lenguaje español y definir una sintaxis y un autómata que le diese soporte, para que de esta forma se pudiera construir al sistema de nomenclatura de datos.

5.3.4.- Revisión sintáctica del lenguaje español.

A través de este análisis, se propone identificar las características que debería tener el lenguaje de nomenclatura de datos, en base al lenguaje español y que siga la filosofía del lenguaje OF.

En el lenguaje español, todo **enunciado bimembre (oración)** está compuesto por un **sujeto** y un **predicado.** El sujeto puede ser un **morfema autónomo** como él, yo, tú etc. o un **sintagma:** la luz del sol; pero el predicado siempre lo constituye un sintagma

El **sintagma sujeto** siempre tiene como núcleo a un **sustantivo.** Cualquier sintagma cuyo núcleo sea un sustantivo se denomina **"sintagma sustantivo".**

El núcleo sustantivo puede ir sólo o acompañado de **modificadores.**

El núcleo del predicado puede también llevar modificadores, pero esto no ocurre siempre.

El núcleo de cualquier sintagma sustantivo puede llevar estos modificadores:

modificador directo: se une al núcleo por sí mismo.

Por ejemplo: cielo azul

cubierto de nubes

modificador indirecto: se une al núcleo mediante un nexo o encabezador.

Por ejemplo: limón de calidad
sin semilla

Se define al **sustantivo** de la siguiente forma:

Sintácticamente: es la única unidad lingüística que funciona como núcleo de sujeto, aposición, objeto directo o indirecto.

Semánticamente: sirve para nombrar a todos los individuos de la misma especie si se trata de sustantivos comunes como: perro, casa, hombre o para nombrar a un ser.

El **adjetivo** se define de la siguiente forma:

Sintácticamente: es el modificador directo del sustantivo.

Semánticamente: expresa una cualidad o determinación del sustantivo.

Construcciones sustantivas:
Son sustantivos que se forman de varias palabras relacionadas entre si. P.ej.:

El sombrero de tres picos.
El número de teléfono del empleado.

De lo anteriormente explicado, es claro que el lenguaje de nomenclatura de datos deberia comportarse en forma muy similar a lo descrito por los párrafos anteriores. Es decir, la **definición de un dato debe tener como mínimo al sujeto de una oración.** Este sujeto debe tener entonces la estructura de una construcción sustantiva, en la que pueden existir forzozamente un núcleo sustantivo (que corresponderá a la palabra clase) y uno o varios modificadores directos o indirectos, los cuales corresponderán a las palabras entidad y modificadores.

Por lo anterior, se presenta a continuación un análisis más profundo sobre los sintagmas nominales sujetos:

Sintagma nominal sujeto:

A veces un morfema única funciona como sujeto: Yo juego. Pero cuando el sujeto está representado por un sintagma, puede constar de:

- Un núcleo nominal (un sustantivo)
- Un núcleo nominal y modificadores

Los **modificadores del sintagma nominal (sujeto)** se clasifican en 2 tipos: Modificador directo y modificador indirecto.

- Modificador directo:

Un modificador directo se une por si mismo a su núcleo,

ninguna otra palabra sirve de enlace entre ellas. P.ej.: El libro azul. Nombre completo.

- Modificador indirecto:

Un modificador indirecto se relaciona con su nucleo mediante un nexo o particula encabezadora. Los modificadores indirectos que se unen a su núcleo mediante una **preposición** se denominan **complementos.** En un complemento, lo que sigue a la preposición recibe el nombre de **término.** P.ej.: número **de teléfono.**

- Construcción adjetiva:

Toda palabra que funcione como modificador directo del sustantivo desempeña el oficio de adjetivo. El adjetivo puede, a su vez, tener modificadores directos o indirectos (complementos).

Un núcleo adjetivo con sus modificadores forma una construcción adjetiva.

- Complemento adjetivo:

El automóvil de lujo — el automóvil lujoso
núcleo compl. — núcleo adj.

La construcción "de lujo" es un modificador indirecto del núcleo sustantivo automóvil. Tiene como encabezador a la preposición "de". Se trata de un complemento. El complemento "de lujo" puede sustituirse por el adjetivo "lujoso". Es un

complemento subordinado a un núcleo sustantivo que funciona como adjetivo.

Los modificadores del sintagma nominal sujeto pueden modificar a cualquier sintagma sustantivo, no importa la función que éste desempeñe.

De lo expresado anteriormente, la construcción que se considera apropiada para el lenguaje de nomenclatura de datos, corresponde al de un núcleo nominal acompañado por una serie de modificadores, en donde el núcleo nominal corresponderá a la palabra clase.

5.3.5.- Gramática del lenguaje de nomenclatura de datos.

A continuación se muestra las reglas de producción que forman parte de la gramática que origina al lenguaje de nomenclatura de datos usado en el sistema.

```
1.1)  <definición> ::=
          <sujeto><complemento_nominal><punto_final>
2.1)  <sujeto> ::=
          <núcleo_nominal><modificador_del_sujeto>
2.2)     |<núcleo_nominal>
3.1)  <núcleo_nominal> ::=
          <artículo><palabra_clase>
3.2)     |<palabra_clase>
4.1)  <modificador_del_sujeto> ::=
          <adjetivo>
4.2)     |<preposición><núcleo_modificador>
4.3)     |<contracción><núcleo_modificador>
5.1)  <núcleo_modificador> ::=
          <sustantivo>
5.2)     |<sustantivo><adjetivo>
6.1)  <complemento_nominal>::=
          <núcleo_entidad>
6.2)     |<núcleo_entidad><complemento_descripción>
7.1)  <núcleo_entidad> ::=
          <preposición><artículo><nucleo_palabra_entidad>
7.2)     |<contracción><núcleo_palabra_entidad>
8.1)  <núcleo_palabra_entidad> ::=
          <palabra_entidad>
```

8.2) |<palabra_entidad><adjetivo>

9.1) <complemento_descripción>::=
<modif_del_complemento>

9.2) |<modif_del_complemento><complemento_descripción>

10.1) <modif_del_complemento> ::=
<preposición><núcleo_modificador>

10.2) |<preposición><artículo><núcleo_modificador>

10.3) |<contracción><núcleo_modificador>

11.1) <palabra_clase> ::= <sustantivo>

12.1) <palabra_entidad> ::= <sustantivo>

13.1) <sustantivo> ::= <identificador>

14.1) <adjetivo> ::= <identificador>

15.1) <identificador> ::= <letra><letra>*

16.1) <artículo> ::= el |la |los |las

17.1) <preposición> ::= a |ante |bajo |cabe |con
|contra |de |desde |en |entre
|hacia |hasta |para |por |que
|según |sin |so |sobra |tras

18.1) <contracción> ::= al |del

19.1) <punto_final> ::= .

20.1) <letra> ::= a |b |c |d |e |f |g |h |i |j
|k |l |m |n |o |p |q |r |s |t
|u |v |w |x |y |z |n |ñ |-
|á |é |í |ó |ú
|A |B |C |D |E |F |G |H |I |J
|K |L |M |N |O |P |Q |R |S |T
|U |V |W |X |Y |X |N |Ñ |-

A continuación se muestra la explicación de cada una de las reglas de producción del lenguaje de nomenclatura de datos:

Las primeras reglas de producción a considerar, fueron aquellas que permitieran construir al **núcleo de la definición** del dato, es decir, estructuras que tuviesen la forma de la **palabra clase**, o bien, **un artículo y la palabra clase**. Estas reglas de producción son: (nota: la numeración de las reglas de producción, corresponde a la lista anteriormente mostrada).

3.1) <núcleo_nominal> ::=
<artículo><palabra_clase>

3.2) |<palabra_clase>

Con estas reglas de producción, pueden formarse los siguientes ejemplos:

el número, el nombre, la fecha, número, fecha, texto

Ahora bien, el **núcleo nominal** puede estar acompañado (o no) de un **modificador** que brinde mayor información sobre la palabra clase. Si la palabra clase es de tipo "modificador forzoso", es necesaria la presencia de éste. De lo contrario, su presencia es opcional. Para el primer caso, el **modificador del núcleo nominal** se define mediante la siguiente regla de producción:

4.1) <Modificador_del_sujeto> ::=
<adjetivo>

Por ejemplo, un modificador podría ser el adjetivo "secreto".

Sin embargo, el **modificador del sujeto** no se restringe a ser un adjetivo, sino que puede ser formado por la presencia de una frase modificadora. Como ejemplos, pueden citarse los siguientes: "de nacimiento", "de acceso secreto".
Este tipo de estructuras se definen mediante las siguientes reglas de producción:

```
4.2) <modificador_del_sujeto> ::=
          <preposición><núcleo_modificador>
4.3)       |<contracción><núcleo_modificador>
5.1) <núcleo_modificador> ::=
          <sustantivo>
5.2)       |<sustantivo><adjetivo>
```

Ahora bien, el **núcleo nominal** y el **modificador del sujeto**, forman en conjunto al **sujeto** de la definición del dato. Las siguientes reglas de producción indican lo anterior:

```
2.1) <sujeto> ::=
          <núcleo_nominal><modificador_del_sujeto>
2.2)       |<núcleo_nominal>
```

La regla 2.1 se aplica para el caso de que la palabra clase corresponda al tipo "palabra clase de modificador forzoso". Para el caso de palabras clase de tipo "palabra clase de modificador opcional", se aplican ambas reglas.

En base a las reglas de producción hasta ahora descritas, es posible construir los siguientes ejemplos:

nombre

fecha de contratacion

número de acceso secreto

número confidencial

Hasta este momento, faltan describir reglas de producción que permitan indicar la presencia de la **palabra entidad** asi como de otras **palabras modificadoras** en la definición del dato. Para indicar esta presencia, se define a la siguiente regla:

1.1) <definición> ::=
<sujeto><complemento_nominal><punto_final>

En esta regla el <punto_final> es el caracter "." , que es usado para indicar el fin del texto de definicion del dato a nombrar. Lo anterior es reflejado mediante la siguiente regla de producción:

19.1) <punto_final> ::= .

El **complemento_nominal** engloba entonces a la **palabra entidad** y a los **modificadores** en la forma en que lo indican las siguientes reglas de producción:

6.1) <complemento_nominal> ::=
<núcleo_entidad>

6.2) |<núcleo_entidad><complemento_descripción>

Es decir, la regla 6.1 se usa para indicar la presencia de la **palabra entidad** sin modificadores, mientras que la regla 6.2 permite especificar una serie de modificadores que complementen la definición del dato.

Para poder relacionar la **palabra entidad** con la **palabra clase,** se requiere de la presencia de un **artículo** y una **preposición** o una **contracción.** Lo anterior es especificado a través de las siguientes reglas de producción:

7.1) <núcleo_entidad> ::=
<preposición><artículo><núcleo_palabra_entidad>

7.2) |<contracción><núcleo_palabra_entidad>

La presencia del **núcleo_palabra_entidad,** indica que la palabra entidad podría presentarse sola o acompañada por un adjetivo, tal y como lo definen las siguientes reglas de producción:

8.1) <núcleo_palabra_entidad> ::=
<palabra_entidad>

8.2) |<palabra_entidad><adjetivo>

Con las reglas de producción que definen al **complemento nominal**, pueden construirse los siguientes ejemplos:

de el cliente .

del empleado .

de cuenta personal .

y con las reglas de produccion que definen a la **definición** del dato, pueden construirse los siguientes ejemplos:

número de acceso secreto de el cliente .

fecha de contratación del empleado .

fecha de apertura de la cuenta personal .

nombre del cotitular .

Ahora bien, sólo falta describir a las reglas de producción que permitan indicar la presencia de **modificadores** que enriquezcan la definición del dato a nombrar. Estos modificadores podrían estar formados por un **sustantivo** o por un **sustantivo** seguido por un **adjetivo**, tal y como lo muestra las siguientes reglas de producción:

5.1) <núcleo_modificador> ::=
<sustantivo>

5.2) |<sustantivo><adjetivo>

Por ejemplo: caja permanente

Estos modificadores podrían ser una lista de sustantivos y adjetivos, los cuales se relacionan entre si a través de la presencia de preposiciones y artículos. Para lo anterior, se usa a las siguientes reglas de producción:

10.1) <modif_del_complemento> ::=
<preposición><núcleo_modificador>

10.2) |<preposición><artículo><núcleo_modificador>

10.3) |<contracción><núcleo_modificador>

Así, con estas reglas pueden construirse los siguientes ejemplos:

a la caja permanente

de banca electrónica

Ahora bien, la cantidad de **modificadores del complemento** puede ser variable. Por lo tanto, se requiere de la siguiente regla de producción recursiva:

9.1) <complemento_descripción> ::=
<modif_del_complemento>

9.2) |<modif_del_complemento><complemento_descripción>

La regla 9.1 define la presencia de un sólo modificador, mientras que la regla 9.2 permite la presencia de un número ilimitado de modificadores. Con las reglas anteriores, pueden obtenerse los siguientes ejemplos:

de acceso secreto a cajas permanentes

de uso selecto de cuenta maestra

Con la regla de producción 6.1 pueden relacionarse estos modificadores con la palabra clase y la palabra entidad, dando como resultado a definiciones completas. Por ejemplo:

a) número del cliente para acceso secreto a las cajas permanentes .

b) fecha de contratación del empleado .

c) fecha de vencimiento del pagaré contratado con el banco.

Ahora bien, tanto la **palabra clase** como la **palabra entidad** deben ser sustantivos. Esto es importante, ya que el sistema debe hacer la validación de los gramemas característicos de los sustantivos para aceptar palabras del lenguaje español en la definición del dato que se desea nombrar. Lo anterior se indica mediante la existencia de las siguientes reglas de producción:

11.1) <palabra_clase> ::= <sustantivo>
12.1) <palabra_entidad> ::= <sustantivo>

A su vez, tanto los **sustantivos** como los **adjetivos** son considerados como **identificadores**, los que en realidad solo se reducen a ser una secuencia de letras. Lo anterior es expresado mediante las siguientes reglas de producción:

13.1) <sustantivo> ::= <identificador>
14.1) <adjetivo> ::= <identificador>
15.1) <identificador> ::= <letra><letra>*

La siguiente regla de producción muestra los **articulos** válidos en el sistema:

16.1) <artículo> ::= el |la |los |las

La siguiente regla de producción muestra la lista de **preposiciones** válidas en el sistema:

17.1) <preposición> ::= a |ante |bajo |cabe |con |contra
|de |desde |en |entre |hacia |hasta
|para |por |que |según |sin |so
|sobra |tras

La siguiente regla de producción muestra la lista de **contracciones** válidas en el sistema:

18.1) <contracción> ::= al |del

El **alfabeto del lenguaje de nomenclatura de datos** es muy semejante al alfabeto del lenguaje español. Sin embargo, la letras "ch" y "ll" se consideran como la concatenación de las letras "c","h" y "l","l", respectivamente. La letra "ñ" si se le considera como parte del alfabeto del lenguaje de nomenclatura de datos. Las letras acentuadas también se consideran como parte del lenguaje, sin embargo, las rutinas encargadas del manejo del teclado convierten las letras acentuadas a letras no acentuadas. El sistema automáticamente convierte letras mayúsculas a minúsculas. No forman parte del alfabeto los caracteres numéricos ni los caracteres de puntuación especial. El único caracter especial que forma parte del alfabeto es: "-" . El caracter "." también forma parte del alfabeto, y se usa para indicar el final del texto de definición. Lo anterior está representado por:

20.1) <letra> ::= a |b |c |d |e |f |g |h |i |j
|k |l |m |n |o |p |q |r |s |t
|u |v |w |x |y |z |ñ |-
|á |é |í |ó |ú
|A |B |C |D |E |F |G |H |I |J
|K |L |M |N |O |P |Q |R |S |T
|U |V |W |X |Y |X |Ñ |-

19.1) <punto_final> ::= .

5.3.6.- Autómata del sistema de nomenclatura de datos para la aplicación del lenguaje de nomenclatura de datos.

La intención de esta sección consiste en mostrar al autómata que se desarrolló para el sistema. Este autómata tiene el propósito de ayudar al administrador de datos a aplicar el lenguaje de nomenclatura de datos, de forma tal que la definición del dato esté correctamente estructurada. Para ello, el autómata indica en cada estado, los elementos del lenguaje que deberían especificarse, de acuerdo con la gramática mostrada en la sección anterior.

El **estado #0** pide que se especifique un **<artículo>** o **<palabra clase>**. Dependiendo de lo que escriba el usuario, el autómata cambiará de estado: Si se trata de un **<artículo>**, cambiará al estado =1, mientras que si se especifica una **<palabra clase>** cambiará al estado =2.

El **estado #1** pide que se especifique una **<palabra clase>** . Si la palabra que se escribe en este estado es una **<palabra clase>**, entonces el autómata cambiará al estado #2.

El **estado #2** pide que se especifique un **<adjetivo>** que califique a la palabra clase o una **<preposición>**. Dependiendo de lo que especifique el usuario, el autómata cambiará de estado: Si especifica un **<adjetivo>** el autómata cambiará al estado #5, mientras que si especifica una **<preposición>** cambiará al estado #3.

El **estado #3** pide que se especifique un **<sustantivo>** que modifique a la palabra clase, o bien, que se especifique directamente a la **<palabra_entidad>**. Para que el autómata cambie de estado, deberá revisar el tipo de la palabra clase que se haya especificado en la definición del dato. Esto es necesario, pues dependiendo del tipo de palabra clase realizará las siguientes acciones:

Si la **palabra clase es de tipo "modificador forzoso"**, entonces el autómata sólo cambiará al estado #3 considerando a la palabra escrita como un **sustantivo**. En este caso, si el usuario escribe una palabra entidad, entonces el sistema indicará que es un error y ayudará al usuario a corregirlo. Esto desencadena la ejecución de otro autómata, el cual formará una cadena de ayudas al usuario.

Si la **palabra clase es de tipo "modificador opcional"**, entonces el sistema revisará si la palabra escrita es una **palabra entidad**. Si tal es el caso, entonces el autómata cambiará al estado #8. Si no es palabra entidad, sino un **sustantivo**, entonces cambiará el autómata al estado #4.

El **estado #4** pide que se especifique un **<adjetivo>** que forme parte del modificador de la palabra clase, o bien que se especifique una **<preposición>**. Si el usuario especifica un **<adjetivo>**, el autómata cambiará al estado #5, mientras

que si se especifica una **<preposición>** entonces cambiará al estado #6.

El **estado #5** pide que se especifique una **<preposición>**. Al especificarse una **<preposición>**, el autómata cambiará al estado #7.

El **estado #6** pide que se especifique un **<artículo>** o una **<palabra entidad>**. Si se escribe un **<artículo>**, el autómata cambiará al estado #7. Si se escribe una **<palabra entidad>**, entonces cambiará al estado #8.

El **estado #7** pide que se especifique una **<palabra_ entidad>**. Al especificarse una **<palabra_entidad>**, entonces el autómata cambiará al estado #8.

El **estado #8** pide que se especifique un **<adjetivo>** que califique a la palabra entidad, o una **<preposición>** o al **<punto final>**. Si se escribe un **<adjetivo>** el autómata cambiará al estado #9. Si se escribe una **<preposición>** el autómata cambiará al estado #11. Si se escribe una **<contracción>** el autómata cambiará al estado #12. Si se escribe el **<punto final>** el autómata cambiará al estado #10.

El **estado #9** pide que se especifique una **<preposición>** o **<punto final>**. Si se escribe una **<preposición>** el autómata cambiará al estado #11. Si se escribe una **<contracción>** el autómata cambiará al estado #12. Si se escribe el **<punto final>** el autómata cambiará al estado #10.

El estado #10 es el estado final del autómata. En él, se considera como completa y bien estructurada la definición del dato.

El estado #11 pide que se especifique un **<articulo>** o un **<sustantivo>** que sirva como modificador. Si se escribe un **<articulo>** el autómata cambiará al estado #12. Si se escribe un **<sustantivo>** entonces cambiará al estado #13.

El estado #12 pide que se especifique un **<sustantivo>** que sirva como modificador. Al escribir un **<sustantivo>** el autómata cambiará al estado #13.

El estado #13 pide que se especifique un **<adjetivo>** que forme parte del modificador, o una **<preposición>** o el **<punto final>**. Si se escribe un **<adjetivo>** el autómata cambiará al estado #9. Con una **<preposición>** cambiará al estado #11. Con una **<contracción>** cambiará al estado #12. Finalmente, con el **<punto final>** cambiará al estado #10.

En base a los estados y elementos del lenguaje que se especifiquen en dichos estados, el autómata es capaz de ayudar a seguir la sintaxis del lenguaje de nomenclatura de datos. Para mostrar que el autómata realmente da soporte al lenguaje, se requiere seguir la siguiente notación:

Cada estado requiere que se especifique un elemento del lenguaje para que el autómata cambie de dicho estado (estado

predecesor) a otro estado (estado sucesor). Lo anterior se indica con la siguiente notación:[1]

<transición> ::= (# de estado predecesor, elemento del lenguaje, # de estado sucesor)

Por ejemplo, para indicar que en el estado #0 se requiere de un <artículo> para que el autómata cambie al estado #1, se escribiría:

(0,<artículo>,1)

Con esta notación, es posible indicar una secuencia de transiciones del autómata: Para ello, debe cumplirse que el estado sucesor de la (n-1)-ésima transición sea igual al estado predecesor de la n-ésima transición.

<secuencia> ::= <lista_de_transiciones>

<lista_de_transiciones> ::=

<transición>

|<transición><lista_de_transiciones>

Por ejemplo, si se quiere indicar la secuencia formada por un artículo escrito en el estado #0 y la palabra clase escrita en el estado #1, siguiendo esta notación se escribiría:

(0,<artículo>,1)(1,<palabra_clase>,2)

---

1 Esta notación está inventada, no es formal, ni se tomó como referencia ningún libro o artículo de revista.

De este ejemplo se nota que el estado sucesor de la primera transición (estado #1) es igual al estado predecesor de la segunda transición.

Para simplificar secuencias, una secuencia puede asociársele un identificador, de forma tal que si se hacer referencia a dicha secuencia se escribiría su identificador:

<identificación_de_secuencia> ::=

(<identificador>) : <secuencia>

Esta notación puede usarse también para indicar la concatenación de secuencias, para ello, bastará escribir los identificadores de secuencias en una lista.

Considerando esta notación, puede explicarse entonces la forma en que el autómata da soporte a las reglas de producción del lenguaje de nomenclatura de datos:

La siguiente secuencia:

(núcleo_nominal_1) :
(0,<artículo>,1)(1,<palabra clase>,2)

da soporte a la siguiente regla de producción:

3.1) <núcleo_nominal> ::=
<artículo><palabra_clase>

La siguiente secuencia:

(núcleo_nominal_2) :
(0,<palabra clase>,2)

da soporte a la siguiente regla de producción:

3.2) <núcleo_nominal> ::=

<palabra_clase>

Como puede verse, las secuencias (núcleo_nominal_1) y (nucleo_nominal_2) son totalmente equivalentes, por lo que se les llamará (núcleo_nominal) para poder hacer referencia de ellas más adelante.

La siguiente secuencia:

(modif_del_sujeto_1) :
(2,<adjetivo>,5)

da soporte a la siguiente regla de producción:

4.1) <modificador_del_sujeto> ::=
<adjetivo>

Mientras que la siguiente secuencia:

(núcleo_modificador_1) :
(3,<sustantivo>,4)

da soporte a la siguiente regla de producción:

5.1) <núcleo_modificador> ::=
<sustantivo>

La siguiente secuencia:

(núcleo_modificador_2) :
(núcleo_modificador_1)(4,<adjetivo>,5)

da soporte a la siguiente regla de producción:

5.2) <núcleo_modificador> ::=
<sustantivo><adjetivo>

Las siguientes secuencias:

(modif_del_sujeto_2) :
(2,<preposición>,3)(núcleo_modificador_1)

(modif_del_sujeto_3) :
(2,<preposición>,3)(núcleo_modificador_2)

dan soporte a la siguiente regla de producción:

4.2) <modificador_del_sujeto> ::=
<preposición><núcleo_modificador>

Las siguientes secuencias:

(modif_del_sujeto_4) :
(2,<contracción>,3)(núcleo_modificador_1)

(modif_del_sujeto_5) :
(2,<contracción>,3)(núcleo_modificador_2)

dan soporte a la siguiente regla de producción:

4.3) <modificador_del_sujeto> ::=
<contracción><núcleo_modificador>

La siguiente secuencia:

(núcleo_nominal)

corresponde a la siguiente regla de producción:

2.2) <sujeto> ::= <núcleo_nominal>

Mientras que las secuencias:

(sujeto_1) : (núcleo_nominal)(modif_del_sujeto_1)
(sujeto_2) : (núcleo_nominal)(modif_del_sujeto_2)
(sujeto_3) : (núcleo_nominal)(modif_del_sujeto_3)
(sujeto_4) : (núcleo_nominal)(modif_del_sujeto_4)
(sujeto_5) : (núcleo_nominal)(modif_del_sujeto_5)

dan soporte a la siguiente regla de producción:

2.1) <sujeto> ::=
<núcleo_nominal><modificador_del_sujeto>

Las siguientes secuencias:

(palabra_entidad_1) : (3,<palabra_entidad>,8)
(palabra_entidad_2) : (6,<palabra_entidad>,8)
(palabra_entidad_3) : (7,<palabra_entidad>,8)

dan soporte a la siguiente regla de producción:

8.1) <núcleo_palabra_entidad> ::=

<palabra_entidad>

Como puede observarse, las secuencias anteriores son equivalentes entre sí, por lo que se les llamará (palabra_entidad) para hacer referencia a ellas más adelante.

De esta forma, la secuencia:

(núcleo_palabra_entidad_1) :
(palabra_entidad)

da soporte a la siguiente regla de producción:

8.1) <núcleo_palabra_entidad> ::= <palabra_entidad>

Mientras que la siguiente secuencia:

(núcleo_palabra_entidad_2) :
(palabra_entidad)(8,<adjetivo>,9)

da soporte a la siguiente regla de producción:

8.2) <núcleo_palabra_entidad> ::=
<palabra_entidad><adjetivo>

Las siguientes secuencias:

(núcleo_entidad_1) :
(4,<preposición>,6)(6,<artículo>,7)
(núcleo_palabra_entidad_1)

(núcleo_entidad_2) :
(4,<preposición>,6)(6,<artículo>,7)
(núcleo_palabra_entidad_2)

(núcleo_entidad_3) :
(5,<preposición>,6)(6,<artículo>,7)
(núcleo_palabra_entidad_1)

(núcleo_entidad_4) :
(5,<preposición>,6)(6,<artículo>,7)
(núcleo_palabra_entidad_2)

dan soporte a la siguiente regla de producción:

7.1) <núcleo_entidad> ::=
<preposición><artículo><núcleo_palabra_entidad>

Las siguientes secuencias:

(núcleo_entidad_5) :
(4,<contracción>,7)(núcleo_palabra_entidad_1)

(núcleo_entidad_6) :
(4,<contracción>,7)(núcleo_palabra_entidad_2)

(núcleo_entidad_7) :
(5,<contracción>,7)(núcleo_palabra_entidad_1)

(núcleo_entidad_8) :
(5,<contracción>,7)(núcleo_palabra_entidad_2)

dan soporte a la siguiente regla de producción:

7.2) <núcleo_entidad> ::=
<contracción><núcleo_palabra_entidad>

Ahora bien, con la siguientes secuencias:

(núcleo_entidad_1)(8,<punto_final>,10)
(núcleo_entidad_2)(9,<punto_final>,10)
(núcleo_entidad_3)(8,<punto_final>,10)
(núcleo_entidad_4)(9,<punto_final>,10)
(núcleo_entidad_5)(8,<punto_final>,10)
(núcleo_entidad_6)(9,<punto_final>,10)
(núcleo_entidad_7)(8,<punto_final>,10)
(núcleo_entidad_8)(9,<punto_final>,10)

se da soporte a la siguientes reglas de producción:

6.1) <complemento_nominal> ::=
<núcleo_entidad>

Al cumplirse la regla de producción, se está cumpliendo entonces la siguiente regla, pues se está especificando al <punto final> en el texto:

1.1) <definición> ::=
<sujeto><complemento_nominal><punto_final>

Las siguientes secuencias:

(núcleo_modif_3) :
(11,<sustantivo>,13)

(núcleo_modif_4) :
(12,<sustantivo>,13)

dan soporte a la siguiente regla de producción:

5.1) <núcleo_modificador> ::= <sustantivo>

y las siguientes secuencias:

(núcleo_modif_5) :
(11,<sustantivo>,13)(13,<adjetivo>,9)

(núcleo_modif_6) :
(12,<sustantivo>,13)(13,<adjetivo>,9)

dan soporte a la siguiente regla de produccion:

5.2) <núcleo_modificador> ::= <sustantivo><adjetivo>

Las siguientes secuencias:

(8,<preposición>,11)(núcleo_modif_3)
(8,<preposición>,11)(núcleo_modif_5)
(9,<preposición>,11)(núcleo_modif_3)
(9,<preposición>,11)(núcleo_modif_5)

dan soporte a la siguiente regla de producción:

10.1) <modif_del_complemento> ::=
<preposición><núcleo_modificador>

Las siguientes secuencias:

(8,<preposición>,11)(11,<artículo>,12)(núcleo_modif_4)
(8,<preposición>,11)(11,<artículo>,12)(núcleo_modif_6)
(9,<preposición>,11)(11,<artículo>,12)(núcleo_modif_4)
(9,<preposición>,11)(11,<artículo>,12)(núcleo_modif_6)

dan soporte a la siguiente regla de producción:

10.2) <modif_del_complemento> ::=
<preposición><artículo><núcleo_modificador>

Las siguientes secuencias:

(8,<contracción>,12)(núcleo_modif_4)
(8,<contracción>,12)(núcleo_modif_6)

(9,<contracción>,12)(núcleo_modif_4)
(9,<contracción>,12)(núcleo_modif_6)

dan soporte a la siguiente regla de producción:

10.3) <modif_del_complemento> ::=
<contracción><núcleo_modificador>

Asimismo, las secuencias anteriores dan entonces soporte a la siguiente regla de producción:

9.1) <complemento_descripción> ::=
<modif_del_complemento>

Por último, resta analizar la siguiente regla de producción recursiva, la cual permite incluir un número indeterminado de modificadores en la definición del dato a nombrar:

9.2) <complemento_descripción> ::=
<modif_del_complemento><complemento_descripción>

Para que esta regla de producción sea soportada por el autómata, se requiere que se hubieran de antemano presentado las secuencias anteriormente descritas para soportar a las reglas 10.1, 10.2 y 10.3. En estos casos, el autómata se encontraría en cualquiera de los siguientes estados:

Estado #9 o estado #13.

Para que se presente la recursividad, deben presentarse cualquiera de las siguientes transiciones:

(9,<contracción>,12)
(9,<preposición>,11)
(13,<adjetivo>,9)
(13,<preposición>,11)
(13,<contracción>,12)

Por el contrario, la recursividad termina al presentarse cualquiera de las siguientes transiciones:

(9 ,<punto_final>,10)
(13,<punto_final>,10)

Al presentarse estas transiciones, en realidad se está cumpliendo la siguiente regla de produccion:

6.2) <complemento_nominal> ::=
<núcleo_entidad><complemento_descripción>

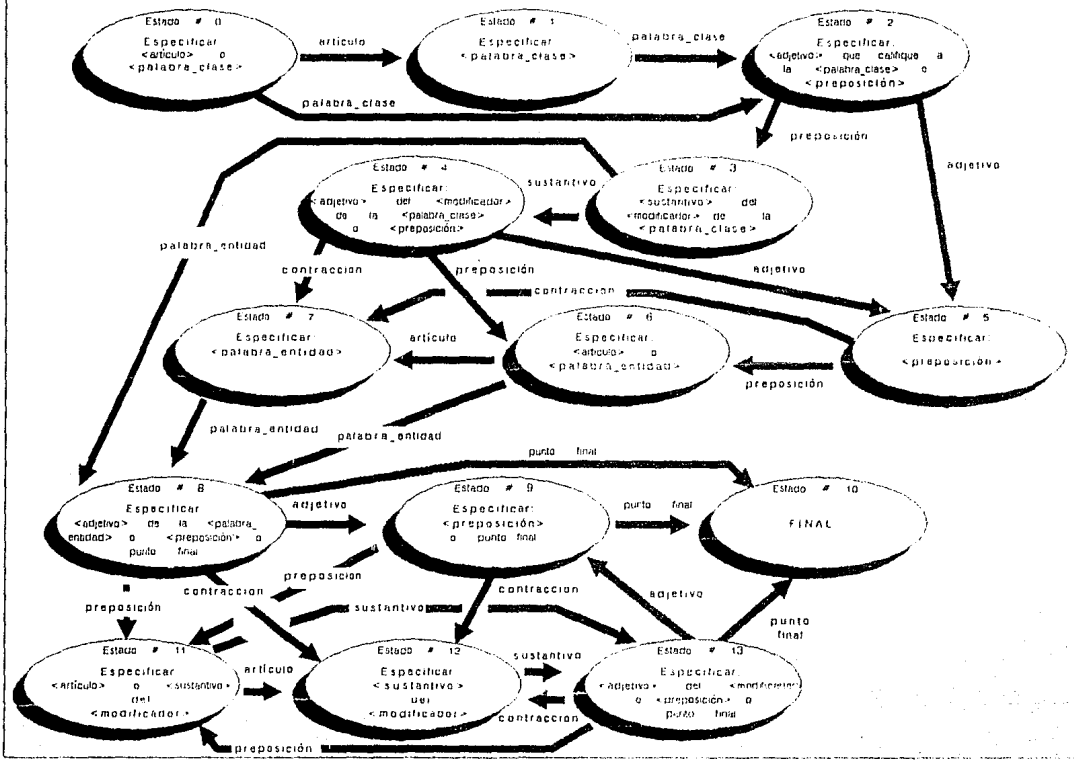
Al cumplirse esta regla, entonces inmediatemente se cumple la siguiente regla (debido a la presencia del <punto_final>, que hace que el automata alcance el estado final):

1.1) <definición> ::=
<sujeto><complemento_nominal><punto_final>

Por último, un autómata puede representarse en forma de gráfica dirigida. Esta gráfica se encuentra formada por nodos y arcos. Los nodos representan a los estados del autómata, mientras que los arcos representan los elementos del lenguaje que se requieren en cada nodo para hacer una transición de un nodo (o estado) a otro. En el extremo inicial de un arco se encuentra el nodo (o estado) antecesor de la transición, mientras que en el extremo final del arco se encuentra el nodo (o estado) sucesor de la transición.

En la siguiente pagina se muestra la grafica que representa al autómata descrito en esta sección.

ESTADOS PRINCIPALES PARA LA AYUDA EN LA APLICACION DEL LENGUAJE DE NOMENCLATURA DE DATOS.



5.3.7.- Análisis morfológico en el sistema de nomenclatura de datos.

A través del autómata descrito en la sección anterior, el sistema es capaz de ayudar al administrador de datos a aplicar el lenguaje de nomenclatura de datos. Sin embargo, al revisar las reglas de producción del lenguaje, podrá notarse que tanto las **palabras clase, palabras entidad, sustantivos,** así como los **adjetivos,** se reducen a una sóla categoría sintáctica. Esta categoría corresponde a la de **identificadores,** los cuales solo son una secuencia de caracteres alfabéticos en cualquier orden.

Lo anterior lleva a considerar que se requiere de un acervo de palabras que realmente pertenezcan al lenguaje. Sin embargo, llevar el control de un diccionario de palabras es realmente difícil cuando el número de palabras es grande. Por esta razón, se consideró que debería investigarse alguna característica común entre los sustantivos del lenguaje español, (así como los adjetivos) que permitiera identificarlos dentro de una serie de letras encadenadas al azar.

Teniendo en consideración lo anterior, se determinó que la forma más viable de hacer el análisis anterior, sería a través del estudio de los morfemas de las palabras del lenguaje español. A través de los morfemas también es posible determinar si las palabras que se están escribiendo

se encuentran en singular o plural, masculino o femenino. De esta forma, a través del conocimiento de reglas de construcción del género y número de las palabras, es posible pasar una palabra del masculino al femenino (o viceversa), o bien, del plural al singular.

Con este análisis, el sistema alcanza un nivel de inteligencia que le permite identificar si una cadena de caracteres realmente es una palabra del lenguaje español (sustantivo o adjetivo) y determinar su género y número. Además, el manejo de un diccionario de palabras se simplifica, ya que este diccionario sólo estaría compuesto por aquellas palabras que son excepciones a las reglas identificadas en este análisis.

A continuación se presenta el análisis sobre los gramemas del lenguaje español y su uso en el sistema de nomenclatura de datos:

Analícese la siguiente oración:

"Compré este libro."

En la oración anterior, la palabra **"compré"** está integrada por dos unidades: **"compr"** **"é"**

"compr" significa adquirir algo pagando un precio convenido.

"é" indica que el hablante es el comprador y que el hecho mencionado ya ocurrió.

En esta palabra no se pueden hallar unidades más simples que tengan significado. Estas unidades significantes mínimas se llaman **morfemas.** Los morfemas aún se pueden descomponer en

otras unidades mas simples, pero que no tienen significado propio, que se conocen como **fonemas.**

Los fonemas son **unidades sonoras** que pertenecen al sistema de un lenguaje. En el lenguaje español que se habla en México, existen 22 fonemas. Los fonemas son elementos sonoros y se representan en la lengua escrita por medio de letras, por lo que algunos fonemas se representan con varias letras. P.ej: el fonema /s/ esta representado por las letras s,c,z.

Algunos morfemas pertenecen a inventarios indeterminados y abiertos y su significado se refiere a la realidad. Estos morfemas se llaman **lexemas.** Por ejemplo:

mes-a libr-o libr-eria camin-o camin-ante

Otros morfemas pertenecen a inventarios reducidos y cerrados, es decir, con pocas posibilidades de cambio. Indican la categoria gramatical o expresan las relaciones de las palabras, reciben el nombre de **grammemas.** Algunos grammemas van sueltos como : yo, tú, ayer, de , etc. pero la mayor parte acompaña a los lexemas. Un mismo lexema aparece en palabras de diferente categoria:

sustantivo	verbo	adjetivo
clar-idad	**aclar**-ar	**clar**-o
am-or	**am**-ar	**am**-able

De esta forma, los grammemas se encargan de identificar la categoria sintáctica de las palabras, permitiendo identificar si una palabra corresponde a un **sustantivo, adjetivo, verbo, adverbio,** etc.

El estudio de los lexemas permiten conocer la significación y ortografía de las palabras. Todas las palabras que tienen un mismo lexema pertenecen a la misma familia. Las palabras de una misma familia se llaman **afines** y suelen escribirse del mismo modo. Esto es de gran utilidad para la ortografía. Los lexemas experimentan pocos cambios ortográficos.

En el idioma español, los lexemas van acompañados de gramemas. Hay gramemas impuestos por las demás palabras del contexto, su uso es **obligatorio.** Los gramemas del sustantivo y del adjetivo deben significar el mismo género y el mismo número. Esta repetición obligatoria de gramemas se llama **concordancia.**

Existen otros gramemas cuyo uso no es exigido por el contexto, sino que el hablante está en libertad de usarlos cuando le convenga. Son gramemas no obligatorios. P. ej. gramemas de **aumentativos, diminutivos, colectivos, sustantivos abstractos, etc.**

Los gramemas de concordancia caracterizan a: sustantivos, adjetivos y verbos. Los gramemas de concordancia del sustantivo y adjetivo se llaman **gramemas nominales: género y número.**

Los gramemas **de género** permiten clasificar los sustantivos y adjetivos como **masculinos (o),** o **femeninos (a).** Ej:

alumn-**o** alumn-**a**

Algunos sustantivos carecen de grafemas de género. Son masculinos si atraen adjetivos masculinos. Son femeninos si atraen adjetivos femeninos. Los adjetivos que no tienen grafemas de género, indiferentemente acompañan a sustantivos masculinos o femeninos.

Para saber el género de un sustantivo:

1.- se le antepone el artículo.

Son masculinos si admiten **el** o **los**.

Son femeninos si admiten **la** o **las**.

Son neutros si admiten **lo**.

2.- los sustantivos que comienzan por **a** o **ha** acentuadas llevan **el** aunque sean femeninos. P.ej. el agua, el hambre.

Para formar el femenino de un sustantivo:

Se cambia el grafema **o** por **a** o se añade el grafema **a** al masculino. Algunas palabras tienen terminaciones especiales: **esa, isa, ina, triz.**

Hay femeninos que se forman con otra palabra de distinto lexema. P.ej. caballo - yegua.

Los **grafemas de número** permiten clasificar los sustantivos y adjetivos como **singulares o plurales.**

Para conocer con seguridad el número de los sustantivos

se les antepone el artículo: Cuando se les puede poner delante **el, la, lo** , están en singular; si admiten los o las pertenecen al **plural.** El número de los adjetivos se determina con el del sustantivo que los acompaña.

Los sustantivos y adjetivos suelen seguir las siguientes reglas para formar su plural:

Forman el plural con el gramema **-s** :

Los sustantivos y adjetivos terminados en vocal sin acento: mesas, buenos, coches, yaquis, etc.

Los terminados en **é** : cafés, corsés, etc.

Forman el plural con el gramema **-es**:

Los sustantivos y adjetivos terminados en vocal acentuada menos en é: rajáes, alelíes, bantúes...

Los terminados en consonante:

pared --> paredes mármol --> mármoles

Excepciones: papás, sofás, mamás, dominós.

Plurales especiales:

Sustantivos terminados en z, cambian la "z" en "c" y añaden "es".

Existen otros gramemas cuyo uso no es exigido por el contexto, sino que el hablante está en libertad de usarlos cuando le convenga. Son gramemas **no obligatorios.** Entre éstos, se encuentran los **gramemas facultativos.**

Los **gramemas facultativos** no son de concordancia, su uso no es obligatorio. Modifican el significado de los lexemas formando palabras nuevas. Los gramemas facultativos se clasifican en : **prefijos e infijos.**

Los **prefijos** son gramemas facultativos que van antes del lexema. Los **infijos** son gramemas facultativos que van después del lexema.

Clases de prefijos:

- **Prefijos cuantitativos:**

  Añaden al lexema una significación de cantidad.

  P.ej. **re-staurar semi-automático mono-cultivo.**

- **Prefijos cualitativos:**

  No se refieren a la cantidad sino a la calidad.

  P.ej. **equi-distante multi-familiar**

- **Prefijos autónomos:**

  A veces forman palabras, funcionan como gramemas sueltos . Aqui entran las preposiciones, y otros son por ejemplo: anti, bi, cent, circum, equi, extra, inter, preter, super, trans, tri, ultra, vice, archi, cata, deca, epi, foto, hecto, hiper, hipo kilo, meta, miria, mono, neo, proto, seudo.

Clases de infijos:

- **Infijos cuantitativos:**

  Se refieren a los **diminutivos, aumentativos, intensivos y despectivos:**

- **diminutivos:**

  -ill-, -it-, -ic-, -in-, -cill-, -cit-, -cic-, -et-, -ecill-, -ecit-, -ecic-, -ececill-, -ececit-, -ececic-.

- **aumentativos:**

  -ón-, -ot-, -zón-, -az-, -ozón-.

- **intensivos:**

  -isim-, -érrim-.

- **despectivos:**

  -ac-, -uc-, -uch-, -ej-, -ij-, -orri-, -orr-, -alla, -uza.

Los **infijos** modifican el sentido de los lexemas. De esta forma, existen infijos que son característicos de sustantivos y otros que lo son de adjetivos. A continuación se muestran las listas de estos dos casos:

Infijos que forman sustantivos:

Infijos	significados	ejemplos
-ada, -aje -al, -ar -edo, -eda -men, -ario	colectivo	bandada, equipaje colmenar arboleda herbario velamen
-ería, -ada	acción	monería bufonada
-er-	recipiente	salero
-ado, -ato -azgo	dignidad	licenciado, liderazgo
-ía, -io, -er-	profesión	ingeniería

-ante,-ista,-ari-	oficio	estudiante, notario
-azo	efecto producido por un objeto	martillazo
-ia, -ica, -ina ina, -ato, -uro	voces técnicas	carburo, carbonato,
-ada, -ata, -ido -ida,-io,-mento -miento	accion verbal	levantamiento estallido
-ción -ión -mer -or -ura	acción y su efecto	vejación, reunión gravamen escozor
-ismo	sistema o doctrina	catolicismo capitalismo
-monio	estado, resultado	matrimonio patrimonio
-or	agente	matador, inspector
-ancia, -encia -eza, -ia, -ie -anza, -ad,-ez -monia, -tud, -umbre, -ura	cualidades abstractas	vagancia, creencia grandeza, miseria plenitud, cordura mansedumbre

infijos que forman adjetivos:

infijos	significados	ejemplos:
-ad-, -eñ-, -er- -u-, -al, -ante, -ente,-id-,-ari- -el-, -ient-,-ici-	cualidad	fatigado, trigueño asiduo,abitual, reverente afligido novel, avariento
-áce- , -ad-,	semejanza	rosáceo leonado

-ent- -os-	plenitud	virulento, sabroso
-ud-	posesión despectivo	trompudo
-al, -ar,-ici- -ieg-, -iz,-in- -und-	propensión	mortal, militar andariego, quebradizo iracundo
-áne-,-e-,-ern-	pertenencia	cutáneo
-estre, -este, in-, -i- -iz-, -un- -esc-	relación	campestre, celeste, campesino, tardío fronterizo vacuno principesco
-en-, -er-, -ésim-	orden	onceno, primero
-ón, -bund-	intensidad	mandón nauseabundo
-iv-	capacidad de actuar	curativo, imaginativa
-ble	que puede recibir la acción	digerible, voluble amable, indeleble
-ici-, -il	aptitud, modo de ser	servil
-er-	cualidad pasiva o activa	asadero
-bre	que produce algo	salubre
-átil, -az, -or, -triz	que produce la acción	volátil, voraz, inyector, ganador, retardatriz.
-ori-	que sirve para la accion	declaratorio.

En base a los infijos mostrados anteriormente, así como de las reglas definidas para la construcción del plural y singular de sustantivos y adjetivos, se definen las listas de gramemas característicos. Estas listas se usarán para generar al autómata que se encargará de analizar a las palabras escritas en el texto de definición de los datos a nombrar:

sng=singular plr=plural

msc=masculino fmn=femenino

***=indet

Principales gramemas de sustantivos:

Morfemas	número y género
-ica	sng,fmn
-ada	sng,fmn
-eda	sng,msc
-ida	sng,fmn
-ia	sng,fmn
-ia	sng,fmn
-ancia	sng,fmn
-encia	sng,fmn
-monia	sng,fmn
-eria	sng,fmn
-ura	sng,fmn
-ata	sng,fmn
-ista	sng,***
-eza	sng,fmn
-anza	sng,fmn
-ad	sng,fmn
-tud	sng,fmn
-ie	sng,***
-aje	sng,msc
-umbre	sng,fmn
-ante	sng,***
-al	sng,msc
-men	sng,msc
-ión	sng,fmn

-ción	sng,fmn
-ado	sng,msc
-edo	sng,msc
-ido	sng,msc
-azgo	sng,msc
-io	sng,msc
-monio	sng,msc
-ario	sng,msc
-ismo	sng,msc
-uro	sng,msc
-ato	sng,msc
-miento	sng,msc
-mento	sng,msc
-azo	sng,msc
-ar	sng,msc
-or	sng,msc
-ez	sng,fmn

Principales grammemas de adjetivos:

Gramema	número y género
-ble	sng,***
-bre	sng,***
-estre	sng,***
-ante	sng,***
-ente	sng,***
-este	sng,***
-iz	sng,***
-al	sng,***
-il	sng,***
-átil	sng,***
-ón	sng,msc
-ar	sng,***
-or	sng,***
-az	sng,***

Principales grammemas de género y número:

-a	sng,fmn,***
-ina	sng,fmn,sus
-esa	sng,fmn,sus
-isa	sng,fmn,sus
-o	sng,msc,***
-s	plr,***,***
-es	prl,msc,sus
-triz	sng,fmn,sus

En base al análisis anterior, el sistema se encarga de tomar ciertas acciones, tal y como se explican a continuación:

Cuando el sistema analice si una palabra corresponde a un sustantivo o a un adjetivo, se debe considerar que entrará a funcionar un autómata. Este autómata analizará las letras terminales (del final al principio de la palabra) y determinará si se trata de un adjetivo, un sustantivo o de una palabra que no corresponde ninguno de estos dos tipos. Para ello, el autómata se construye a partir de las listas de gramemas característicos de sustantivos y adjetivos.

Ahora bien, puede darse el caso de que al dar de alta una nueva palabra en el acervo, ésta se escriba en plural. Podría suceder que esta misma palabra ya existiera en el acervo, pero en singular. El sistema debe ser capaz de detectar el plural de cada palabra y generar su singular, de forma que se evite almacenar una misma palabra con diferentes gramemas de número y género.

Al momento de construir el nombre del dato con las abreviaturas de las palabras claves, debe considerarse que las reglas de abreviación podrían indicar un cambio en los gramemas de número y género. Es decir, para generar la abreviatura de una palabra, ésta debe ser independiente del género y número de la palabra.

Por estas razones, el sistema debe ser capaz de identificar si una palabra es sustantivo o adjetivo, al mismo tiempo de identificar su género y número.

Para ello, el sistema aplica las siguientes reglas:

1) Cualquier palabra que se escriba, deberá determinarse si se encuentra en plural o singular. Si se encuentra en plural, el sistema la cambiará internamente a singular y posteriormente analizará si es un adjetivo o sustantivo en base a las listas definidas en el punto 2).

   Para determinar si una palabra se encuentra en plural o singular, así como para cambiar de plural a singular, el sistema aplica las siguientes reglas:

   1.1) Si la última letra corresponde a una s, se analizan los siguientes casos:

   1.2) La penúltima letra corresponde a una vocal no acentuada, expecto la e, o la penúltima letra corresponde a la é acentuada, entonces se desecha la última s de la palabra.

   1.3) La penúltima letra corresponde a una e, entonces podría tratarse del gramema es. Se analiza entonces que la antepenúltima letra corresponda a una constante o a una vocal acentuada, excepto la

e. Si tal es el caso, entonces se desecha al gramema **es**.

1.4) Si se ha identificado al gramema **es**, y la antepenúltima letra es una **c**, entonces se desecha al gramema **es** y la letra **c** se cambia por la letra **z**.

1.5) Si no se aplican los casos anteriores, entonces la palabra analizada no corresponde al lenguaje español.

1.6) Si la palabra analizada no termina en **s**, entonces se considera que se encuentra en singular.

2) Una vez que el sistema establece que la palabra a analizar se encuentra en singular, entonces aplica el análisis de gramemas a través de un autómata.

El autómata de análisis de gramemas se construye a partir de los siguientes gramemas caracteristicos:

	sng=singular	plr=plural
	msc=masculino	fmn=femenino
***=indet	adj=adjetivo	sus=sustantivo

Principales de sustantivos:

Morfemas	número y género	morfema invertido
-ica	sng,fmn,sus	aci-
-ada	sng,fmn,sus	ada-
-eda	sng,msc,sus	ade-
-ida	sng,fmn,sus	adi-
-ia	sng,fmn,sus	ai-
-ia	sng,fmn,sus	ai- o ai-
-ancia	sng,fmn,sus	aicna-
-encia	sng,fmn,sus	aicne-
-monia	sng,fmn,sus	ainom-
-eria	sng,fmn,sus	aire- o aire-
-ura	sng,fmn,sus	aru-
-ata	sng,fmn,sus	ata-
-ista	sng,***,sus	atsi-
-eza	sng,fmn,sus	aze-
-anza	sng,fmn,sus	azna-
-ad	sng,fmn,sus	da-
-tud	sng,fmn,sus	dut-
-ie	sng, ,sus	ei-
-aje	sng,msc,sus	eja-
-umbre	sng,fmn,sus	erbmu-
-ante	sng,***,sus	etna-
-al	sng,msc,sus	la-
-men	sng,msc,sus	nem-
-ión	sng,fmn,sus	nói- o noi-
-ción	sng,fmn,sus	nóic- o noic-
-ado	sng,msc,sus	oda-
-edo	sng,msc,sus	ode-
-ido	sng,msc,sus	odi-
-azgo	sng,msc,sus	ogza-
-io	sng,msc,sus	oi-
-monio	sng,msc,sus	oinom-
-ario	sng,msc,sus	oira-
-ismo	sng,msc,sus	omsi-
-uro	sng,msc,sus	oru-
-ato	sng,msc,sus	ota-
-miento	sng,msc,sus	otneim-
-mento	sng,msc,sus	otnem-
-azo	sng,msc,sus	oza-
-ar	sng,msc,sus	ra-
-or	sng,msc,sus	ro-
-ez	sng,fmn,sus	ze-

Principales de adjetivos:

-ble	sng,***,adj	elb-
-bre	sng,***,adj	erb-
-estre	sng,***,adj	ertse-
-ante	sng,***,adj	etna-
-ente	sng,***,adj	etne-

-este	sng,***,adj	etse-		
-iz	sng,***,adj	zi-		
-al	sng,***,adj	la-		
-il	sng,***,adj	li-		
-átil	sng,***,adj	litá-	o	lita-
-ón	sng,msc,adj	no-	o	no-
-ar	sng,***,adj	ra-		
-or	sng,***,adj	ro-		
-az	sng,***,adj	za-		

En estas listas, se incluye la columna de "gramema invertido", esto se debe a que como los gramemas se encuentran a la derecha (parte terminal) de las palabras, es posible entonces invertir el orden de las letras que forman a las palabras para poder analizarlas mediante el autómata mencionado en esta sección.

5.4.- Diseño lógico del sistema de nomenclatura de datos:

5.4.1.- Red semántica del sistema de nomenclatura de datos:

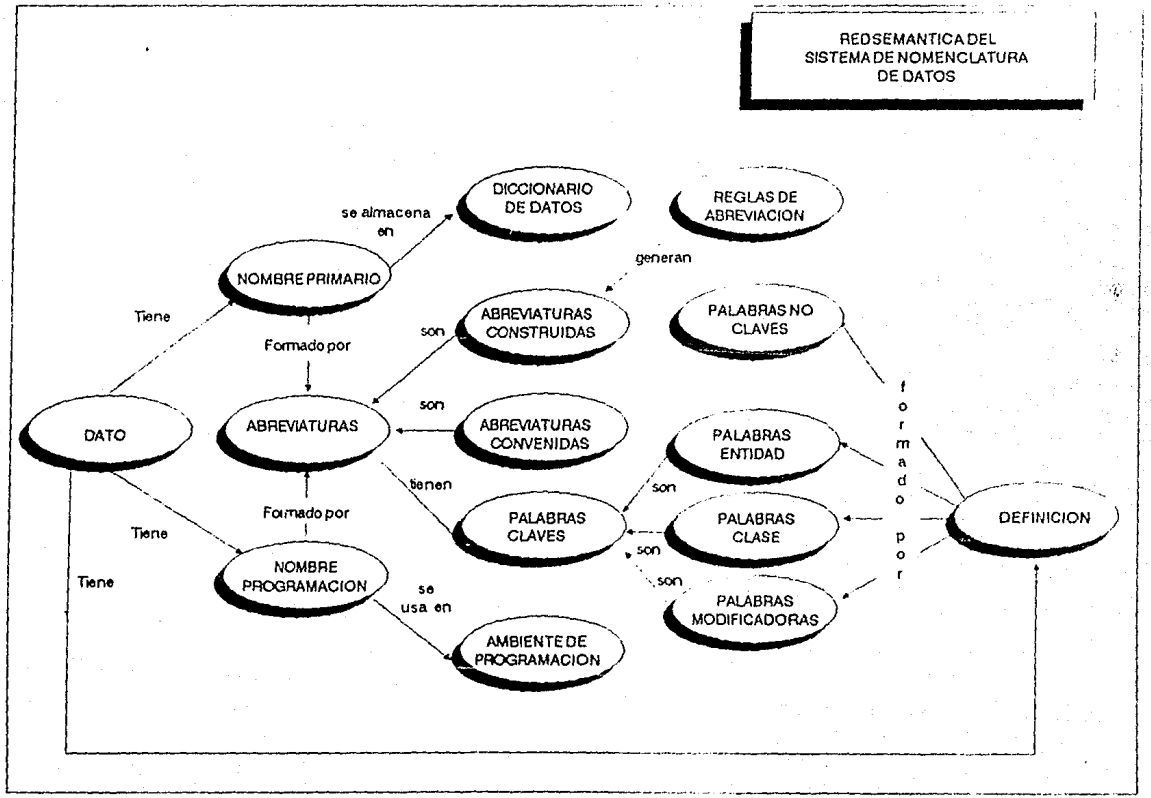
A través de la siguiente red semántica, se trata de mostrar en forma resumida, el conocimiento que se consideró necesario en el diseño del sistema de nomenclatura de datos.

De esta forma, a través de esta red semántica se indica que cada dato tiene asociada una definición única, la cual está formada por una palabra clase, una palabra entidad y por una o varias palabras modificadoras, además de palabras no claves (artículos y preposiciones).

La nomenclatura de datos considera a la palabra clase, palabra entidad y palabras modificadoras como palabras claves, las cuales tienen asociadas una serie de abreviaturas. Estas abreviaturas pueden ser construidas a partir de un conjunto de reglas de abreviación, o bien, corresponder a abreviaturas convenidas en la institución. Ahora bien, estas reglas de abreviación podrán ser distintas en diferentes ambientes de programación o en distintos diccionarios de datos.

A partir de las abreviaturas de las palabras claves de la definición del dato, se construyen el nombre primario y nombres de programación que están asociados al dato.

La siguiente página muestra la red semántica descrita en esta sección.


REDSEMANTICA DEL
SISTEMA DE NOMENCLATURA
DE DATOS
DICCIONARIO
DE DATOS
REGLAS DE
ABREVIACION
se almacena
en
generan
NOMBRE PRIMARIO
Tiene
Formado por
ABREVIATURAS
CONSTRUIDAS
PALABRAS NO
CLAVES
son
DATO
ABREVIATURAS
son
ABREVIATURAS
CONVENIDAS
PALABRAS
ENTIDAD
tienen
son
Formado por
Tiene
PALABRAS
CLAVES
PALABRAS
CLASE
DEFINICION
formado por
Tiene
NOMBRE
PROGRAMACION
son
son
se
usa en
PALABRAS
MODIFICADORAS
AMBIENTE DE
PROGRAMACION

5.4.3.- Descripción del autómata usado para la aplicación del lenguaje de nomenclatura de datos:

A continuación se muestra la descripción de todos y cada uno de los estados que forman al autómata que da seguimiento a la aplicación del lenguaje de nomenclatura de datos.

Para cada estado, se muestran:

- Elemento del lenguaje recibido en el estado anterior.
- Elemento del lenguaje esperado en el estado actual.
- Transición al siguiente estado.

- Descripción del estado: error
  - Elemento recibido en el estado anterior:
    error detectado en la descripción
    ejecuta automata_para_manejo_de_errores
    mat_sintaxis[-1,0]:=edo_anterior;

- Descripción del estado: cero
  - Elemento esperado en el estado actual:
    <artículo> o <palabra_clase> de la descripción
    ejecuta autómata_para_manejo_de_menus
  - Transición al siguiente estado:
    sem:=semántica(artículo o palabra_clase)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: uno
  - Elemento recibido en el estado anterior:
    <artículo>
  - Elemento esperado en el estado actual:
    <palabra_clase> de la descripción

- Transicion al siguiente estado:
  sem:=semántica(palabra_clase)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: dos

  - Elemento recibido en el estado anterior:
    <palabra_clase>

  - Elemento esperado en el estado actual:
    <adjetivo> que califique a la <palabra_clase> o
    <preposición>

  - Transición al siguiente estado:
    sem:=semántica(adjetivo o preposición)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: tres

  - Elemento recibido en el estado anterior:
    si (dato='del')o(dato='al')
          -> <preposición> + <articulo>
    si no,   <preposición>

  - Elemento esperado en el estado actual:
    <sustantivo> del <modificador> de la <palabra_clase>
       si la palabra clase es:
       de modificador opcional
    -> o <palabra_entidad> de la descripción

  - Transición al siguiente estado:
    sem:=semántica(sustantivo de modificador)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: cuatro

  - Elemento recibido en el estado anterior:
    <sustantivo>

  - Elemento esperado en el estado actual:
    <adjetivo> del <modificador> de la <palabra_clase> o
    <preposición>

  - Transición al siguiente estado:
    sem:=semántica(adjetivo o preposición)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: cinco

- Elemento recibido en el estado anterior:
  <adjetivo>

- Elemento esperado en el estado actual:
  <preposición>

- Transición al siguiente estado:
  sem:=semántica(preposición)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: seis

- Elemento recibido en el estado anterior:
  <preposición>

- Elemento esperado en el estado actual:
  <artículo> o <palabra_entidad> de la descripción

- Transición al siguiente estado:
  sem:=semántica(artículo o palabra_entidad)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: siete

- Elemento recibido en el estado anterior:
  <artículo>
  si el estado anterior fue 4 ó 5
  -> <preposición> + <artículo>

- Elemento esperado en el estado actual:
  <palabra_entidad>

- Transición al siguiente estado:
  sem:=semántica(palabra_entidad)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: ocho

- Elemento recibido en el estado anterior:
  <palabra_entidad>

- Elemento esperado en el estado actual:
  <adjetivo> que califique a la <palabra_entidad>
  o <preposición> o punto final

- Transición al siguiente estado:
  sem:=semántica(adjetivo o preposición o punto_final)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: nueve

- Elemento recibido en el estado anterior:
  <adjetivo>

- Elemento esperado en el estado actual:
  <preposición> o punto final

- Transición al siguiente estado:
  sem:=semántica(preposición o punto_final)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: diez

  Descripción completa

- Descripción del estado: once

  - Elemento recibido en el estado anterior:
    <preposición>

  - Elemento esperado en el estado actual:
    <artículo> o <sustantivo> del <modificador>

  - Transición al siguiente estado:
    sem:=semántica(artículo o sustantivo)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: doce

  - Elemento recibido en el estado anterior:
    <artículo>
    si el estado anterior fue 8,9 ó 13
    -> <preposición> + <artículo>
  - Elemento esperado en el estado actual:
    <sustantivo> del <modificador>

  - Transición al siguiente estado:
    sem:=semántica(sustantivo)
    Edo_sigte::=MTE_sintaxis[estado_actual,sem]

- Descripción del estado: trece

  - Elemento recibido en el estado anterior:
    <sustantivo>

  - Elemento esperado en el estado actual:
    <adjetivo> del <modificador> o
    <preposición> o
    punto final

- Transición al siguiente estado:
  sem:=semántica(adjetivo o preposición o punto)
  Edo_sigte::=MTE_sintaxis[estado_actual,sem]

A continuación se muestra la matriz de transición de estados para el autómata que controla la revisión de la sintaxis del lenguaje de nomenclatura de datos:

De cada elemento de la matriz MTE_sintaxis[i,j]:=k

"i" representa el estado actual del autómata,

"j" representa el resultado de la revisión semántica de el elemento escrito en dicho estado "i".

"k" representa el estado al que se efectúa la transición del estado "i" al estado "k".

```
MTE_sintaxis [0,artículo]          := 1
MTE_sintaxis [0,palabra_clase]     := 2

MTE_sintaxis [1,palabra_clase]     := 2

MTE_sintaxis [2,preposición]       := 3
MTE_sintaxis [2,adjetivo]          := 5
MTE_sintaxis [2,contracción]       := 3

MTE_sintaxis [3,palabra_entidad]   := 8
MTE_sintaxis [3,sustantivo]        := 4

MTE_sintaxis [4,preposición]       := 6
MTE_sintaxis [4,adjetivo]          := 5
MTE_sintaxis [4,contracción]       := 7

MTE_sintaxis [5,preposición]       := 6
MTE_sintaxis [5,contracción]       := 7

MTE_sintaxis [6,artículo]          := 7
MTE_sintaxis [6,palabra_entidad]   := 8

MTE_sintaxis [7,palabra_entidad]   := 8

MTE_sintaxis [8,punto_final]       := 10
MTE_sintaxis [8,preposición]       := 11
MTE_sintaxis [8,adjetivo]          := 9
```

```
MTE_sintaxis [8,contraccion]      := 12

MTE_sintaxis [9,preposición]      := 11
MTE_sintaxis [9,punto_final]      := 10
MTE_sintaxis [9,contraccion]      := 12

MTE_sintaxis [11,sustantivo]      := 13
MTE_sintaxis [11,artículo]        := 12

MTE_sintaxis [12,sustantivo]      := 13

MTE_sintaxis [13,adjetivo]        := 9
MTE_sintaxis [13,punto_final]     := 10
MTE_sintaxis [13,preposición]     := 11
MTE_sintaxis [13,contraccion]     := 12
```

5.4.3.- Reglas para llevar a cabo el análisis sintáctico y semántico de la definición del dato a nombrar:

En el autómata anterior, se utiliza una función de transición de estados, la cual utiliza la evaluación de ciertas reglas usadas para llevar a cabo el análisis sintáctico y semántico.

A continuación se muestran las reglas a seguir para determinar la función de cada palabra escrita en la definición del dato a nombrar

1) artículo o palabra_clase:

```
(
        si_es_artículo     -> semántica::= artículo
si no,  si_es_vacío        -> semántica::= no_va_vacío
si no,  si_es_punto        -> semántica::= no_va_punto
si no,  si_es_preposición  -> semántica::= no_va_preposición
si no,  si_es_pal_clase    -> semántica::= palabra_clase
si no,  si_es_entí_sem     -> semántica::= no_va_entidad
si no,                     -> semántica::= no_es_clase
)
```

2) artículo o sustantivo:

```
(
        si_es_vacío          -> semántica::= no_va_vacío
si no,  si_es_artículo       -> semántica::= artículo
si no,  si_es_punto          -> semántica::= no_va_punto
si no,  si_es_preposición    -> semántica::= no_va_preposición
si no,  si_es_clase_sem      -> semántica::= no_va_clase
si no,  si_es_igual_entid    -> semántica::= no_va_entidad
si no,  si_es_sustantivo     -> semántica::= sustantivo
si no,                       -> semántica::= no_es_sustantivo
)
```

3) sustantivo:

```
(
        si_es_vacío          -> semántica::= no_va_vacío
si no,  si_es_punto          -> semántica::= no_va_punto
si no,  si_es_preposición    -> semántica::= no_va_preposición
si no,  si_es_artículo       -> semántica::= no_va_artículo
si no,  si_es_igual_clase    -> semántica::= no_va_clase
si no,  si_es_clase_sem      -> semántica::= no_va_clase
si no,  si_es_igual_entid    -> semántica::= no_va_entidad
si no,  si_es_sustantivo     -> semántica::= sustantivo
si no,                       -> semántica::= no_es_sustantivo
)
```

4) palabra_clase:

```
(
        si_es_vacío          -> semántica::= no_va_vacío
si no,  si_es_punto          -> semántica::= no_va_punto
si no,  si_es_preposición    -> semántica::= no_va_preposición
si no,  si_es_artículo       -> semántica::= no_va_artículo
si no,  si_es_pal_clase      -> semántica::= palabra_clase
si no,  si_es_enti_sem       -> semántica::= no_va_entidad
si no,  si_es_sustantivo     -> semántica::= no_es_clase
si no,                       -> semántica::= no_es_sustantivo
)
```

5) adjetivo o preposición o contracción:

```
(
        si_es_vacío          -> semántica::= no_va_vacío
si no,  si_es_contracción    -> semántica::= contracción
si no,  si_es_preposición    -> semántica::= preposición
```

```
si no,   si_es_punto        -> semántica::= no_va_punto
si no,   si_es_articulo     -> semántica::= no_va_artículo
si no,   si_es_enti_sem     -> semántica::= no_va_entidad
si no,   si_es_clase_sem    -> semántica::= no_va_clase
si no,   si_es_adjetivo     -> semántica::= adjetivo
si no,                      -> semántica::= no_es_adjetivo
)
```

6) sustantivo_modificador:

```
(
         si_es_vacío        -> semántica::= no_va_vacío
si no,   si_es_punto        -> semántica::= no_va_punto
si no,   si_es_preposición  -> semántica::= no_va_preposición
si no,   si_es_artículo     -> semántica::= no_va_artículo
si no,   si_es_igual_clase  -> semántica::= no_va_clase
si no,   si_es_entidad      ->
         (
          si_es_mod_forzoso -> semántica::= no_va_entidad
          si no,            -> semántica::= palabra_entidad
         )
si no,   si_es_clase_sem    -> semántica::= no_va_clase
si no,   si_es_sustantivo   -> semántica::= sustantivo
si no,                      -> semántica::= no_es_sustantivo
)
```

7) preposición o contracción:

```
(
         si_es_vacío        -> semántica::= no_va_vacío
si no,   si_es_contracción  -> semántica::= contracción
si no,   si_es_preposición  -> semántica::= preposición
si no,   si_es_punto        -> semántica::= no_va_punto
si no,   si_es_artículo     -> semántica::= no_va_artículo
si no,   si_es_enti_sem     -> semántica::= no_va_entidad
si no,   si_es_clase_sem    -> semántica::= no_va_clase
si no,                      -> semántica::= no_es_preposición
)
```

8) artículo o palabra_entidad:

```
(
         si_es_punto        -> semántica::= no_va_punto
si no,   si_es_vacío        -> semántica::= no_va_vacío
si no,   si_es_preposición  -> semántica::= no_va_preposición
si no,   si_es_artículo     -> semántica::= artículo
si no,   si_es_entidad      -> semántica::= palabra_entidad
si no,   si_es_clase_sem    -> semántica::= no_va_clase
```

```
si no,  si_es_sustantivo  -> semántica::= sustantivo
si no,                    -> semántica::= no_es_sustantivo
)
```

9) adjetivo o preposición o contracción o punto:

```
(
        si_es_vacio       -> semántica::= no_va_vacio
si no,  si_es_punto       -> semántica::= punto
si no,  si_es_contracción -> semántica::= contracción
si no,  si_es_preposición -> semántica::= preposición
si no,  si_es_articulo    -> semántica::= no_va_articulo
si no,  si_es_enti_sem    -> semántica::= no_va_entidad
si no,  si_es_clase_sem   -> semántica::= no_va_clase
si no,  si_es_adjetivo    -> semántica::= adjetivo
si no,                    -> semántica::= no_es_adjetivo
)
```

10) palabra_entidad:

```
(
        si_es_vacio       -> semántica::= no_va_vacio
si no,  si_es_punto       -> semántica::= no_va_punto
si no,  si_es_preposición -> semántica::= no_va_preposición
si no,  si_es_articulo    -> semántica::= no_va_articulo
si no,  si_es_entidad     -> semántica::= palabra_entidad
si no,  si_es_clase_sem   -> semántica::= no_va_clase
si no,  si_es_sustantivo  -> semántica::= no_es_entidad
si no,                    -> semántica::= no_es_sustantivo
)
```

11) preposición o contracción o punto:

```
(
        si_es_contracción -> semántica::= contracción
si no,  si_es_preposición -> semántica::= preposicion
si no,  si_es_vacio       -> semántica::= no_va_vacio
si no,  si_es_articulo    -> semántica::= no_va_articulo
si no,  si_es_punto       -> semántica::= punto
si no,  si_es_clase_sem   -> semántica::= no_va_clase
si no,  si_es_enti_sem    -> semántica::= no_va_entidad
si no,                    -> semántica::= no_es_preposición
)
```

La revisión de cada uno de los casos arriba descritos, se hará mediante las acciones determinadas en cada una de las

siguientes funciones:

```
función booleana "si_es_punto"
(
   si la palabra es:
   '.'

       -> si_es_punto::=cierto
   si no, si_es_punto::=falso;
)


función booleana "si_es_pal_clase"
(
    busca si la palabra está en:
    el conjunto de palabras clase

    -> si_es_pal_clase::=cierto
    si no,
    (
       convierte la palabra a singular
       busca si la palabra está en:
       el conjunto de palabras clase

           -> si_es_pal_clase::=cierto
       si no, si_es_pal_clase::=falso
     )
)


función booleana "si_es_clase_sem"
(
   busca si la palabra está en:
   el conjunto de palabras clase

   -> si_es_clase_sem::= cierto
   si no,
     (
        arma conjunto de palabras clase
        semejantes a la buscada
        si la cantidad de semejantes es:
        = 1

            -> si_es_clase_sem::= cierto
        si no, si_es_clase_sem::= falso
      )
)


 función booleana "si_es_entidad"
```

```
(
  busca si la palabra está en:
  el conjunto de palabras entidad

  -> si_es_entidad::=cierto
  si no,
    (
      convierte la palabra a singular
      busca si la palabra está en:
      el conjunto de palabras entidad

      -> si_es_entidad::=cierto
      si no, si_es_entidad::=falso
     )
)


función booleana "si_es_enti_sem"
(
  busca si la palabra está en:
  el conjunto de palabras entidad

  -> si_es_enti_sem::= cierto
  si no,
    (
      arma conjunto de palabras entidad
      semejantes a la buscada
      si la cantidad de semejantes es:
      = 1

          -> si_es_enti_sem::= cierto
      si no, si_es_enti_sem::= falso
    )
)


función booleana "si_es_artículo"
(
   si la palabra está en:
   el conjunto de artículos

       -> si_es_artículo ::= cierto
   si no, si_es_artículo ::= falso
)


función booleana "si_es_contracción
(
   si la palabra es:
   'del' o 'al'
```

```
      -> si_es_contracción ::= cierto
  si no, si_es_contracción ::= falso
)


función booleana "si_es_preposicion"
(
   si la palabra está en:
   el conjunto de preposiciones

      -> si_es_preposición ::= cierto
  si no, si_es_preposición ::= falso
)


función booleana "si_es_sustantivo"
(
  si el gramema de la palabra está en:
  conjunto de gramemas de sustantivos
      -> si_es_sustantivo ::= cierto
  si no, si_es_sustantivo ::= falso
)


función booleana "si_es_adjetivo"
(
   si el gramema de la palabra está en:
   conjunto de gramemas de adjetivos
      -> si_es_adjetivo ::= cierto
   si no, si_es_adjetivo ::= falso
)


función booleana "si_es_igual_clase"
(
   si la palabra es:
   igual a la palabra clase de la descripción

      -> si_es_igual_clase ::= cierto
   si no, si_es_igual_clase ::= falso
)


función booleana "si_es_igual_entid"
(
   si la palabra es:
   igual a la palabra entidad de la descripción

      -> si_es_igual_entid ::= cierto
```

```
  si no, si_es_igual_entid ::= falso
)


función booleana "si_es_vacio"
(
   si la palabra es:
   ' ' o de longitud=0

      -> si_es_vacio ::= cierto
  si no, si_es_vacio ::= falso
)


función booleana "si_es_mod_forzoso"
(
    si la palabra clase es:
             de tipo modificador forzoso o
    si el singular de la palabra clase es:
             de tipo modificador forzoso

        -> si_es_mod_forzoso ::= cierto
    si no, si_es_mod_forzoso ::= falso
)
```

5.4.4.- Descripción del autómata usado para controlar el manejo de las ventanas del menú principal del sistema.

La descripción de los autómatas del sistema, está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción de los estados: Esta sección contiene la explicacion de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habra de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección se muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción de los estados del autómata:

Estado -17:
  Acciones:
  -17.1) Llama al autómata encargado de controlar los menús para la definición de reportes.
  -17.2) Despliega ventana -46
         Genera reportes
  -17.3) Activa renglón "Genera reportes"

de esta ventana.
-17.4) Brinca:=verdadero.

Estado -16:
Acciones:
-16.1) Llama a la rutina encargada de efectuar consultas a las listas del sistema. Lista consultada: Nombres de ambientes de programación.

Las siguientes acciones sólo se llevarán a cabo si el usuario tiene la autorización apropiada:

-16.2) Recibe el nombre del ambiente del programación que será borrado del acervo del sistema.
-16.3) Si en el punto anterior, se oprimió la tecla ESC entonces no se borra ningún nombre y esta acción termina a las acciones del estado.
-16.4) Si no se oprimió la tecla ESC sino que se especifió un nombre, se busca al nombre del ambiente de programación en el acervo del sistema.
-16.5) Si el nombre especificado no existe en el acervo, entonces se le indica al usuario y se repite la secuencia de acciones a partir de la acción -16.2.
-16.6) Si el nombre especificado sí existe en el acervo, entonces se verifica que realmente se quiere eliminar.
-16.7) Si el usuario responde afirmativamente, entonces se elimina dicho nombre del acervo. Se actualizan las estructuras de datos dinámicas existentes en memoria principal así como los archivos almacenados en disco.
-16.8) Si el usuario responde negativamente, entonces no se elimina dicho nombre del acervo.

Las siguientes acciones se efectúan aún cuando el usuario no haya tenido autorización para efectuar las acciones anteriores:

-16.9) Elimina todas las ventanas desplegadas en el monitor.
-16.10) Despliega ventana -41.
Altas y Actualizaciones
Eliminaciones
-16.11) Activa renglón "Eliminaciones" de esta ventana.
-16.12) Despliega ventana -42.
Palabras clase
Palabras entidad
Sinónimos
Diccionarios de datos
Ambientes de programación
-16.13) Activa renglón "Ambientes de programación" de esta ventana.

Estado -15:

Acciones:

-15.1) Llama a la rutina encargada de efectuar consultas a las listas del sistema. Lista consultada: Nombres de diccionarios de datos.

Las siguientes acciones sólo se llevarán a cabo si el usuario tiene la autorización apropiada:

-15.2) Recibe el nombre del diccionario de datos que será borrado del acervo del sistema.

-15.3) Si en el punto anterior, se oprimió la tecla ESC entonces no se borra ningún nombre y esta acción termina a las acciones del estado.

-15.4) Si no se oprimió la tecla ESC sino que se especificó un nombre, se busca al nombre del diccionario de datos en el acervo del sistema.

-15.5) Si el nombre especificado no existe en el acervo, entonces se le indica al usuario y se repite la secuencia de acciones a partir de la acción -15.2.

-15.6) Si el nombre especificado si existe en el acervo, entonces se verifica que realmente se quiere eliminar.

-15.7) Si el usuario responde afirmativamente, entonces se elimina dicho nombre del acervo. Se actualizan las estructuras de datos dinámicas existentes en memoria principal así como los archivos almacenados en disco.

-15.8) Si el usuario responde negativamente, entonces no se elimina dicho nombre del acervo.

Las siguientes acciones se efectúan aún cuando el usuario no haya tenido autorización para efectuar las acciones anteriores:

-15.9) Elimina todas las ventanas desplegadas en el monitor.

-15.10) Despliega ventana -41.
Altas y Actualizaciones
Eliminaciones

-15.11) Activa renglón "Eliminaciones" de esta ventana.

-15.12) Despliega ventana -42.
Palabras clase
Palabras entidad
Sinónimos
Diccionarios de datos
Ambientes de programación

-15.13) Activa renglón "Diccionarios de datos" de esta ventana.

Estado -14:

Acciones:

-14.1) Despliega ventana -7
No disponible en esta versión.
para indicar que la opción de eliminación de

sinónimos no está aún disponible.
-14.2) Elimina todas las ventanas desplegadas en el monitor.
-14.3) Despliega ventana -41.
Altas y Actualizaciones
Eliminaciones
-14.4) Activa renglón "Eliminaciones" de esta ventana.
-14.5) Despliega ventana -42.
Palabras clase
Palabras entidad
Sinónimos
Diccionarios de datos
Ambientes de programación
-14.6) Activa renglón "Sinónimos" de esta ventana.

Estado -13:
Acciones:
-13.1) Llama a la rutina encargada de efectuar consultas a las listas del sistema. Lista consultada: palabras entidad.

Las siguientes acciones sólo se llevarán a cabo si el usuario tiene la autorización apropiada:

-13.2) Recibe a la palabra entidad que será borrada del acervo del sistema.
-13.3) Si en el punto anterior, se oprimió la tecla ESC entonces no se borra ningúna palabra y esta acción termina a las acciones del estado.
-13.4) Si no se oprimió la tecla ESC sino que se especificó una palabra, se busca a dicha palabra entidad en el acervo del sistema.
-13.5) Si la palabra especificada no existe en el acervo, entonces se le indica al usuario y se repite la secuencia de acciones a partir de la acción -13.2.
-13.6) Si la palabra especificada si existe en el acervo, entonces se verifica que realmente se quiere eliminar.
-13.7) Si el usuario responde afirmativamente, entonces se elimina dicha palabra del acervo. Se actualizan las estructuras de datos dinámicas existentes en memoria principal así como los archivos almacenados en disco.
-13.8) Si el usuario responde negativamente, entonces no se elimina dicha palabra del acervo.

Las siguientes acciones se efectúan aún cuando el usuario no haya tenido autorización para efectuar las acciones anteriores:

-13.9) Elimina todas las ventanas desplegadas en el monitor.
-13.10) Despliega ventana -41.
Altas y Actualizaciones

Eliminaciones
-13.11) Activa renglón "Eliminaciones" de esta ventana.
-13.12) Despliega ventana -42.
Palabras clase
Palabras entidad
Sinónimos
Diccionarios de datos
Ambientes de programación
-13.13) Activa renglón "Palabras entidad" de esta ventana.

Estado -12:
Acciones:
-12.1) Llama a la rutina encargada de efectuar consultas a las listas del sistema. Lista consultada: palabras clase.

Las siguientes acciones sólo se llevarán a cabo si el usuario tiene la autorización apropiada:

-12.2) Recibe a la palabra clase que será borrada del acervo del sistema.
-12.3) Si en el punto anterior, se oprimió la tecla ESC entonces no se borra ninguna palabra y esta acción termina a las acciones del estado.
-12.4) Si no se oprimió la tecla ESC sino que se especificó una palabra, se busca a dicha palabra clase en el acervo del sistema.
-12.5) Si la palabra especificada no existe en el acervo, entonces se le indica al usuario y se repite la secuencia de acciones a partir de la acción -13.2.
-12.6) Si la palabra especificada sí existe en el acervo, entonces se verifica que realmente se quiere eliminar.
-12.7) Si el usuario responde afirmativamente, entonces se elimina dicha palabra del acervo. Se actualizan las estructuras de datos dinámicas existentes en memoria principal así como los archivos almacenados en disco.
-12.8) Si el usuario responde negativamente, entonces no se elimina dicha palabra del acervo.

Las siguientes acciones se efectúan aún cuando el usuario no haya tenido autorización para efectuar las acciones anteriores:

-12.9) Elimina todas las ventanas desplegadas en el monitor.
-12.10) Despliega ventana -41.
Altas y Actualizaciones
Eliminaciones
-12.11) Activa renglón "Eliminaciones" de esta ventana.
-12.12) Despliega ventana -42.
Palabras clase
Palabras entidad

```
          Sinónimos
          Diccionarios de datos
          Ambientes de programación
 -12.13) Activa renglón "Palabras clase" de esta ventana.

Estado -11:
  Acciones:
  -11.1) Activa_renglón "Ambientes de programación"
         de la ventana desplegada en el estado -6

Estado -10:
  Acciones:
  -10.1) Activa_renglón "Diccionarios de datos"
         de la ventana desplegada en el estado -6

Estado -9:
  Acciones:
  -9.1) Activa_renglón "Sinónimos"
        de la ventana desplegada en el estado -6

Estado -8:
  Acciones:
  -8.1) Activa renglón "Palabras entidad "
        de la ventana desplegada en el estado -6

Estado -7:
  Acciones:
  -7.1) Activa renglón "Palabras clase"
        de la ventana desplegada en el estado -6

Estado -6:
  Acciones:
  -6.1) Despliega ventana -42
          Palabras clase
          Palabras entidad
          Sinónimos
          Diccionarios de datos
          Ambientes de programación
  -6.2) Activa renglón "Palabras clase"
        de esta ventana

Estado -5:
  Acciones:
  -5.1) Activa renglón "Eliminaciones"
        de la ventana desplegada en el estado 1

Estado -4:
  Acciones:
  -4.1) Activa renglón "Altas y cambios"
        de la ventana desplegada en el estado 1

Estado -3:
  Acciones:
  -3.1) Despliega ventana -11
```

```
          Palabras clase
          Palabras entidad
          Reglas de abreviación
          Sinónimos
   -3.2) Activa renglón "Palabras clase"
         de esta ventana.

Estado -2:
  Acciones:
  -2.1) Activa renglón "Fin de sesión"
        de la ventana desplegada en el estado 4

Estado -1:
  Acciones:
  -1.1) Activa renglón "Del menú principal"
        de la ventana desplegada en el estado 4

Estado 0:
  Acciones:
  0.1) Quita todas las ventanas desplegadas en el monitor.
  0.2) Activa la palabra "Ayuda" de la pantalla.
  0.3) Despliega ventana 0
          Artículos
          Preposiciones
          Punto_final
          Palabras_clase
          Palabras_entidad
          Sustantivos
          Adjetivos
  0.4) Activa renglón "Artículos"
       de esta ventana.

Estado 1:
  Acciones:
  1.1) Borra todas las ventanas desplegadas en el monitor.
  1.2) Activa la palabra 'Actualiza' de la pantalla .
  1.3) Despliega ventana -41:
         Altas y cambios
         Eliminaciones
  1.4) Activa renglón "Altas y cambios"
       de esta ventana.

Estado 2:
  Acciones:
  2.1) Borra todas las ventanas desplegadas en el monitor.
  2.2) Activa la frase 'Corregir texto' de la pantalla.
  2.3) Si el estado actual del autómata principal del sistema
       es el estado 0, significa que no se ha escrito palabra
       alguna en la descripción del dato a nombrar. Por la
       razón anterior, no es posible corregir texto, lo cual
       se notifica al usuario. Se modifica la matriz de
       transición de estados del autómata de menús:
       mat_menú[2,2]:=6. De esta forma, si el usuario
       selecciona modificar texto y no existe texto qué
```

modificar, se pasa al estado terminal del presente autómata.

2.4) Si el estado actual del autómata principal del sistema es diferente al estado 0, significa que sí se ha escrito parte de la descripción del dato a nombrar. Se modifica la matriz de transición de estados del autómata de menús: mat_menú[2,2]:=21. De esta forma, si el usuario selecciona modificar texto, se pasa al estado en el cual se llama a la rutina encargada de modificar al texto de descripción del dato a nombrar.

Estado 3:
Acciones:
3.1) Quita todas las ventanas desplegadas en el monitor.
3.2) Activa la palabra "Archivos" de la pantalla.
3.3) Despliega ventana -46
Generación de reportes
3.4) Activa el renglón "Generación de reportes"
de esta ventana.

Estado 4:
Acciones:
4.1) Quita todas las ventanas desplegadas en el monitor.
4.2) Activa la palabra "Salir" de la pantalla.
4.3) Despliega ventana -39
Del menú principal
Fin de sesión
4.4) Activa renglón "Del menú principal"
de esta ventana.

Estado 5:
Acciones:
5.1) Quita todas las ventanas desplegadas en el monitor.
5.2) Activa la palabra "Info" de la pantalla.
5.3) Despliega ventana 11
Autor
Memoria disponible
5.4) Activa renglón "Autor"
de esta ventana.

Estado 6:
Acciones:
6.1) Estado terminal del autómata.
6.2) Quita todas las ventanas desplegadas en el monitor.

Estado 7:
Acciones:
7.1) Activa renglón "Articulos"
de la ventana desplegada en el estado 0

Estado 8:
Acciones:
8.1) Activa renglón "Preposiciones"
de la ventana desplegada en el estado 0

```
Estado 9:
  Acciones:
  9.1) Activa renglón "Punto final"
       de la ventana desplegada en el estado 0

Estado 10:
  Acciones:
  10.1) Activa renglón "Palabras clase"
        de la ventana desplegada en el estado 0

Estado 11:
  Acciones:
  11.1) Activa renglón "Palabras entidad"
        de la ventana desplegada en el estado 0

Estado 12:
  Acciones:
  12.1) Activa renglón "Sustantivos"
        de la ventana desplegada en el estado 0

Estado 13:
  Acciones:
  13.1) Activa renglón "Adjetivos"
        de la ventana desplegada en el estado 0

Estado 14:
  Acciones:
  14.1) Despliega ventana 8 para explicar articulos:
                    ' Ayuda : Articulos '
              'Los articulos válidos en el sistema'
              'son   las    siguientes   palabras:'
              '          el, la, los, las          '
              ' Oprima:  ESC para continuar'
              '           ?  para ayuda extra'

Estado 15:
  Acciones:
  15.1) Despliega ventana -9 para explicar preposiciones:
                    ' Ayuda : Preposiciones '
              'Las preposiciones válidas en el sistema'
              'son las siguientes palabras:'
              '      a,     al,  de,   del,   con,   contra,'
              '      hasta, que, en,   desde, entre, segun, '
              '      sobra, sin, cabe, ante,  bajo,  hacia, '
              '      tras,  por, para                       '
              ' Oprima:  ESC para continuar'
              '           ?  para ayuda extra'

Estado 16:
  Acciones:
  16.1) Construye ventana -10 para explicar la función del
        punto final:
                    ' Ayuda : Punto final '
```

```
            'El punto final sirve para indicar '
            'el fin de la descripción. '
            'Sólo puede usarse en los casos '
            'que indique el sistema. '
            ' Oprima: ESC para continuar'
            '         ? para ayuda extra'
```

Estado 17:
  Acciones:
  17.1) Llama a la rutina encargada de manejar las consultas a las listas del sistema.
        Lista a consultar: Palabras clase.

Estado 18:
  Acciones:
  18.1) Llama a la rutina encargada de manejar las consultas a las listas del sistema.
        Lista a consultar: Palabras entidad.

Estado 19:
  Acciones:
  19.1) Despliega ventana 27 para explicar sustantivos:

```
            ' Ayuda : Sustantivos '
            'Los sustantivos son palabras que modifican'
            'a la palabra clase o a la palabra entidad,'
            'después de haber escrito una preposición'
            'o un artículo.                           '
            ' Oprima: ESC para continuar'
            '         ? para ayuda extra'
```

Estado 20:
  Acciones:
  20.1) Despliega ventana 26 para explicar adjetivos:

```
            ' Ayuda : Adjetivos '
            'Los adjetivos son palabras que modifican'
            'a los sustantivos, indicando alguna cualidad:'
            '     secreto, inválido, válido, completo,'
            ' Oprima: ESC para continuar'
            '         ? para ayuda extra'
```

Estado 21:
  Acciones:
  21.1) Quita todas las ventanas desplegadas en el monitor.
  22.2) Llama a la rutina encargada de corregir el texto de descripción del dato a nombrar.
  22.3) Estado final del autòmata.

Estado 22:
  Acciones:
  22.1) Activa renglón "Palabras clase"
        de la ventana desplegada en el estado -3.

Estado 23:
  Acciones:

23.1) Activa renglon "Palabras entidad"
de la ventana desplegada en el estado -3.

Estado 24:
Acciones:
24.1) Activa renglón "Reglas de abreviación"
de la ventana desplegada en el estado -3.

Estado 25:
Acciones:
25.1) Activa renglón "Sinónimos"
de la ventana desplegada en el estado -3.

Estado 26:
Acciones:
26.1) Llama a la rutina que se encarga del manejo de las consultas a las listas del sistema. Lista consultada: Palabras clase.

Las siguientes acciones se llevarán a cabo sólo si el usuario cuenta con la autorización necesaria:

26.2) Recibe el nombre de la palabra que se desea dar de alta como palabra clase o que se quiere actualizar.
26.3) Si en la acción anterior el usuario oprimió la tecla 'ESC' entonces se da fin a este estado.
26.4) Si en la acción anterior el usuario especificó una palabra, entonces se revisa si esta palabra se encuentra en plural o singular.
26.5) Si la palabra se encuentra en plural, el sistema genera el singular. Indica al usuario que la palabra se encuentra en plural, y que puede cambiarla al singular generado por el sistema o bien, optar por escribir otra palabra.
26.6) Si el usuario opta por escribir otra palabra, se repite la secuencia de acciones a partir de la acción 26.2.
26.7) Si el usuario opta por dejar el plural de la palabra, no hay cambios. Si el usuario opta por el singular de la palabra, entonces se considera el singular generado por el sistema.
26.8) Revisa si el plural o el singular de la palabra especificada (que se espera que sea palabra clase) se encuentra almacenada como palabra entidad. En caso afirmativo, se notifica al usuario que esta palabra ya existe en el acervo del sistema como palabra entidad y por tanto se rechaza la actualización. Se repite la secuencia de acciones a partir de la acción 26.2.
26.9) Revisa si la palabra especificada se encuentra almacenada como preposición. En caso afirmativo, se notifica al usuario que dicha palabra ya existe en el acervo del sistema como preposición y por tanto se rechaza la actualización. Se repite la secuencia de

acciones a partir de la acción 26.2.

26.10) Revisa si la palabra especificada se encuentra almacenada como artículo. En caso afirmativo, se notifica al usuario que la palabra ya existe en el acervo del sistema como artículo y por tanto se rechaza la actualización. Se repite la secuencia de acciones a partir de la acción 26.2.

26.11) Revisa si la palabra especificada se encuentra almacenada en el acervo de palabras clase. En caso negativo, indica al usuario que es una palabra nueva; en caso afirmativo, muestra al usuario la abreviación de esta palabra.

26.12) Recibe la abreviación actualizada de la palabra.

26.13) Si el usuario oprimió la tecla "ESC" entonces esta acción pone fin a este estado.

26.14) Si el usuario especificó una abreviación, entonces se revisa que esta abreviación no esté asociada a otra palabra clase o palabra entidad. En caso afirmativo, se rechaza la actualización y se repite la secuencia de acciones a partir de la acción 26.12.

26.15) En el caso de que se hayan aceptado la palabra clase y su abreviación, entonces se pide al usuario que especifique el tipo de palabra clase: 'De modificador opcional' o 'De modificador forzoso'.

26.16) Recibe el tipo de palabra clase de acuerdo al punto anterior.

26.17) Verifica si el usuario quiere que se actualice el acervo con la palabra clase, abreviación y tipo especificados. En caso afirmativo, se actualizan las estructuras de datos dinámicas existentes en la memoria principal, así como los archivos almacenados en disco.

Las siguientes acciones se llevan a cabo aún cuando el usuario no hubiese tenido autorización para efectuar las acciones anteriores, o si no actualizó al acervo del sistema:

26.18) Quita todas las ventanas desplegadas en el monitor.

26.19) Despliega la ventana -41
Altas y Actualizaciones
Eliminaciones

26.20) Activa renglón "Altas y Actualizaciones" de esta ventana.

26.21) Despliega ventana -11
Palabras clase
Palabras entidad
Reglas de abreviación
Sinónimos

26.22) Activa renglón "Palabras clase" de esta ventana.

Estado 27:
Acciones:

27.1) Llama a la rutina que se encarga del manejo de las consultas a las listas del sistema. Lista consultada: Palabras entidad.

Las siguientes acciones se llevaran a cabo solo si el usuario cuenta con la autorizacion necesaria:

27.2) Recibe el nombre de la palabra que se desea dar de alta como palabra entidad o que se quiere actualizar.
27.3) Si en la acción anterior el usuario oprimió la tecla 'ESC' entonces se da fin a este estado.
27.4) Si en la acción 27.2, el usuario especifico una palabra, entonces se revisa si esta palabra se encuentra en plural o singular.
27.5) Si la palabra se encuentra en plural, el sistema genera el singular. Indica al usuario que la palabra se encuentra en plural, y que puede cambiarla al singular generado por el sistema o bien, optar por escribir otra palabra.
27.6) Si el usuario opta por escribir otra palabra, se repite la secuencia de acciones a partir de la acción 27.2.
27.7) Si el usuario opta por dejar el plural de la palabra, no hay cambios. Si el usuario opta por el singular de la palabra, entonces se considera el singular generado por el sistema.
27.8) Revisa si el plural o el singular de la palabra especificada (que se espera que sea palabra entidad) se encuentra almacenada como palabra clase. En caso afirmativo, se notifica al usuario que esta palabra ya existe en el acervo del sistema como palabra clase y por tanto se rechaza la actualización. Se repite la secuencia de acciones a partir de la accion 27.2.
27.9) Revisa si la palabra especificada se encuentra almacenada como preposición. En caso afirmativo, se notifica al usuario que dicha palabra ya existe en el acervo del sistema como preposición y por tanto se rechaza la actualización. Se repite la secuencia de acciones a partir de la acción 27.2.
27.10) Revisa si la palabra especificada se encuentra almacenada como articulo. En caso afirmativo, se notifica al usuario que la palabra ya existe en el acervo del sistema como articulo y por tanto se rechaza la actualización. Se repite la secuencia de acciones a partir de la acción 27.2.
27.11) Revisa si la palabra especificada se encuentra almacenada en el acervo de palabras clase. En caso negativo, indica al usuario que es una palabra nueva; en caso afirmativo, muestra al usuario la abreviación de esta palabra.
27.12) Recibe la abreviación actualizada de la palabra.
27.13) Si el usuario oprimió la tecla "ESC" entonces esta accion pone fin a este estado.
27.14) Si el usuario especificó una abreviación, entonces se

revisa que esta abreviación no esté asociada a otra palabra clase o palabra entidad. En caso afirmativo, se rechaza la actualización y se repite la secuencia de acciones a partir de la acción 27.12.

27.15) Verifica si el usuario quiere que se actualice el acervo con la palabra entidad y su abreviación especificados. En caso afirmativo, se actualizan las estructuras de datos dinámicas existentes en la memoria principal, así como los archivos almacenados en disco.

Las siguientes acciones se llevan a cabo aún cuando el usuario no hubiese tenido autorización para efectuar las acciones anteriores, o si no actualizó al acervo del sistema:

27.16) Quita todas las ventanas desplegadas en el monitor.
27.17) Despliega la ventana -41
Altas y Actualizaciones
Eliminaciones
27.18) Activa renglón "Altas y Actualizaciones" de esta ventana.
27.19) Despliega ventana -11
Palabras clase
Palabras entidad
Reglas de abreviación
Sinónimos
27.20) Activa renglón "Palabras entidad" de esta ventana.

Estado 28:
Acciones:
28.1) Llama al autómata que controla la definición de reglas de abreviación.
28.2) Quita todas las ventanas desplegadas en el monitor.
28.3) Despliega ventana -41
Altas y cambios
Eliminaciones
28.4) Activa renglón "Altas y cambios"
de esta ventana.
28.5) Despliega ventana -11
Palabras clase
Palabras entidad
Reglas de abreviación
Sinónimos
28.6) Activa renglón "Reglas de abreviación"
de esta ventana.

Estado 29:
Acciones:
29.1) Despliega ventana -7
No disponible en esta versión.

Estado 30:

Acciones:
30.1) Estado terminal del automata.
30.2) Enciende bandera de terminación del automata
principal del sistema.

Estado 31:
Acciones:
31.1) Activa renglón "Autor"
de la ventana desplegada en el estado 5

Estado 32:
Acciones:
32.1) Activa renglón "Memoria disponible"
de la ventana desplegada en el estado 5

Estado 33:
Acciones:
33.1) Despliega ventana con los datos del autor.
33.2) Llama rutina encargada de desplegar la fecha
y hora del sistema.

Estado 34:
Acciones:
34.1) Despliega ventana para presentar memoria disponible.
34.2) Llama rutina encargada de desplegar la memoria
disponible en bytes.

Estado 35:
Acciones:
35.1) Enciende bandera brinca:=verdadero.

Estado 36:
Acciones:
36.1) Quita última ventana desplegada en el monitor.

Estado 37:
Acciones:
37.1) Quita última ventana desplegada en el monitor.

Estado 38:
Acciones:
38.1) Quita última ventana desplegada en el monitor.

Estado 41:
Acciones:
41.1) Quita última ventana desplegada en el monitor.

Estado 42:
Acciones:
42.1) Quita última ventana desplegada en el monitor.

Estado 43:
Acciones:

43.1) Quita última ventana desplegada en el monitor.

Estado 44:
Acciones:
44.1) Quita última ventana desplegada en el monitor.

Estado 45:
Acciones:
45.1) Quita última ventana desplegada en el monitor.

Estado 46:
Acciones:
46.1) Quita última ventana desplegada en el monitor.

Estado 47:
Acciones:
47.1) Quita última ventana desplegada en el monitor.

Función de transición de estados:

```
if not(fin_menu) then
  begin
    if menu_edo_actual<36 then
    begin
      tecla_pas:=trae_tecla;
      if not(brinca) then
        begin
          menu_edo_pasado:=menu_edo_actual;
          menu_edo_sigte:=mat_menu[menu_edo_actual,
                                   tecla_pas];
          menu_edo_actual:=menu_edo_sigte;
        end
      else
        begin
          menu_edo_actual:=menu_edo_pasado;
          menu_edo_sigte:=mat_menu[menu_edo_actual,
                                   tecla_pas];
          menu_edo_actual:=menu_edo_sigte;
          brinca:=false;
        end;
      brinca:=false;
    end
  else
    begin
      menu_edo_pasado:=menu_edo_actual;
      menu_edo_sigte:=mat_menu[menu_edo_actual,
                               tecla_pas];
      menu_edo_actual:=menu_edo_sigte;
    end
end;
```

Matriz de transición de estados de este autómata:

```
for iy:=min_ren_menu to max_ren_menu do
  for ix:=0 to max_col_menu do
     mat_menu[iy,ix]:=35;

     mat_menu[-17,0]:=  4; mat_menu[-17,1]:=2;
     mat_menu[-17,2]:=-17; mat_menu[-17,3]:=6;
     mat_menu[-16,0]:=  2; mat_menu[-16,1]:=0;
     mat_menu[-16,2]:=-16; mat_menu[-16,3]:=47;
     mat_menu[-16,4]:= -7; mat_menu[-16,5]:=-10;

     mat_menu[-15,0]:=  2; mat_menu[-15,1]:=0;
     mat_menu[-15,2]:=-15; mat_menu[-15,3]:=47;
     mat_menu[-15,4]:=-11; mat_menu[-15,5]:=-9;

     mat_menu[-14,0]:=2;   mat_menu[-14,1]:=0;
     mat_menu[-14,2]:=-14; mat_menu[-14,3]:=47;
     mat_menu[-14,4]:=-10; mat_menu[-14,5]:=-8;

     mat_menu[-13,0]:=2;   mat_menu[-13,1]:=0;
     mat_menu[-13,2]:=-13; mat_menu[-13,3]:=47;
     mat_menu[-13,4]:=-9;  mat_menu[-13,5]:=-7;

     mat_menu[-12,0]:=2;   mat_menu[-12,1]:=0;
     mat_menu[-12,2]:=-12; mat_menu[-12,3]:=47;
     mat_menu[-12,4]:=-8;  mat_menu[-12,5]:=-11;

     mat_menu[-11,0]:=2;   mat_menu[-11,1]:=0;
     mat_menu[-11,2]:=-16; mat_menu[-11,3]:=47;
     mat_menu[-11,4]:=-7;  mat_menu[-11,5]:=-10;

     mat_menu[-10,0]:=2;   mat_menu[-10,1]:=0;
     mat_menu[-10,2]:=-15; mat_menu[-10,3]:=47;
     mat_menu[-10,4]:=-11; mat_menu[-10,5]:=-9;

     mat_menu[-9,0]:=2;    mat_menu[-9,1]:=0;
     mat_menu[-9,2]:=-14;  mat_menu[-9,3]:=47;
     mat_menu[-9,4]:=-10;  mat_menu[-9,5]:=-8;

     mat_menu[-8,0]:=2;    mat_menu[-8,1]:=0;
     mat_menu[-8,2]:=-13;  mat_menu[-8,3]:=47;
     mat_menu[-8,4]:=-9;   mat_menu[-8,5]:=-7;

     mat_menu[-7,0]:=2;    mat_menu[-7,1]:=0;
     mat_menu[-7,2]:=-12;  mat_menu[-7,3]:=47;
     mat_menu[-7,4]:=-8;   mat_menu[-7,5]:=-11;

     mat_menu[-6,0]:=2;    mat_menu[-6,1]:=0;
     mat_menu[-6,2]:=-12;  mat_menu[-6,3]:=47;
     mat_menu[-6,4]:=-8;   mat_menu[-6,5]:=-11;

     mat_menu[-5,0]:=2;    mat_menu[-5,1]:=0;
     mat_menu[-5,2]:=-6;   mat_menu[-5,3]:=6;
```

```
mat_menu[-5,4]:=-4;   mat_menu[-5,5]:=-4;

mat_menu[-4,0]:=2;    mat_menu[-4,1]:=0;
mat_menu[-4,2]:=-3;   mat_menu[-4,3]:=6;
mat_menu[-4,4]:=-5;   mat_menu[-4,5]:=-5;

mat_menu[-3,0]:=2;    mat_menu[-3,1]:=0;
mat_menu[-3,2]:=26;   mat_menu[-3,3]:=46;
mat_menu[-3,4]:=23;   mat_menu[-3,5]:=25;

mat_menu[-2,0]:=5;    mat_menu[-2,1]:=3;
mat_menu[-2,2]:=30;   mat_menu[-2,3]:=6;
mat_menu[-2,4]:=-1;   mat_menu[-2,5]:=-1;

mat_menu[-1,0]:=5;    mat_menu[-1,1]:=3;
mat_menu[-1,2]:=6;    mat_menu[-1,3]:=6;
mat_menu[-1,4]:=-2;   mat_menu[-1,5]:=-2;

mat_menu[0,0]:=1;     mat_menu[0,1]:=5;
mat_menu[0,2]:=14;    mat_menu[0,3]:=6;
mat_menu[0,4]:=8;     mat_menu[0,5]:=13;

mat_menu[1,0]:=2;     mat_menu[1,1]:=0;
mat_menu[1,2]:=-3;    mat_menu[1,3]:=6;
mat_menu[1,4]:=-5;    mat_menu[1,5]:=-4;

mat_menu[2,0]:=3;     mat_menu[2,1]:=1;
mat_menu[2,2]:=21;    mat_menu[2,3]:=6;

mat_menu[3,0]:=4;     mat_menu[3,1]:=2;
mat_menu[3,2]:=-17;   mat_menu[3,3]:=6;

mat_menu[4,0]:=5;     mat_menu[4,1]:=3;
mat_menu[4,2]:=6;     mat_menu[4,3]:=6;
mat_menu[4,4]:=-2;    mat_menu[4,5]:=-2;

mat_menu[5,0]:=0;     mat_menu[5,1]:=4;
mat_menu[5,2]:=33;    mat_menu[5,3]:=6;
mat_menu[5,4]:=32;    mat_menu[5,5]:=32;

mat_menu[7,0]:=1;     mat_menu[7,1]:=5;
mat_menu[7,2]:=14;    mat_menu[7,3]:=6;
mat_menu[7,4]:=8;     mat_menu[7,5]:=13;

mat_menu[8,0]:=1;     mat_menu[8,1]:=5;
mat_menu[8,2]:=15;    mat_menu[8,3]:=6;
mat_menu[8,4]:=9;     mat_menu[8,5]:=7;

mat_menu[9,0]:=1;     mat_menu[9,1]:=5;
mat_menu[9,2]:=16;    mat_menu[9,3]:=6;
mat_menu[9,4]:=10;    mat_menu[9,5]:=8;

mat_menu[10,0]:=1;    mat_menu[10,1]:=5;
mat_menu[10,2]:=17;   mat_menu[10,3]:=6;
```

```
mat_menu[10,4]:=11;    mat_menu[10,5]:=9;

mat_menu[11,0]:=1;     mat_menu[11,1]:=5;
mat_menu[11,2]:=18;    mat_menu[11,3]:=6;
mat_menu[11,4]:=12;    mat_menu[11,5]:=10;

mat_menu[12,0]:=1;     mat_menu[12,1]:=5;
mat_menu[12,2]:=19;    mat_menu[12,3]:=6;
mat_menu[12,4]:=13;    mat_menu[12,5]:=11;

mat_menu[13,0]:=1;     mat_menu[13,1]:=5;
mat_menu[13,2]:=20;    mat_menu[13,3]:=6;
mat_menu[13,4]:=7;     mat_menu[13,5]:=12;

mat_menu[14,0]:=1;     mat_menu[14,1]:=5;
mat_menu[14,2]:=36;    mat_menu[14,3]:=36;
mat_menu[14,4]:=36;    mat_menu[14,5]:=36;

mat_menu[15,0]:=1;     mat_menu[15,1]:=5;
mat_menu[15,2]:=37;    mat_menu[15,3]:=37;
mat_menu[15,4]:=37;    mat_menu[15,5]:=37;

mat_menu[16,0]:=1;     mat_menu[16,1]:=5;
mat_menu[16,2]:=38;    mat_menu[16,3]:=38;
mat_menu[16,4]:=38;    mat_menu[16,5]:=38;

mat_menu[17,0]:=1;     mat_menu[17,1]:=5;
mat_menu[17,2]:=39;    mat_menu[17,3]:=39;
mat_menu[17,4]:=39;    mat_menu[17,5]:=39;

mat_menu[18,0]:=1;     mat_menu[18,1]:=5;
mat_menu[18,2]:=40;    mat_menu[18,3]:=40;
mat_menu[18,4]:=40;    mat_menu[18,5]:=40;

mat_menu[19,0]:=1;     mat_menu[19,1]:=5;
mat_menu[19,2]:=41;    mat_menu[19,3]:=41;
mat_menu[19,4]:=41;    mat_menu[19,5]:=41;

mat_menu[20,0]:=1;     mat_menu[20,1]:=5;
mat_menu[20,2]:=42;    mat_menu[20,3]:=42;
mat_menu[20,4]:=42;    mat_menu[20,5]:=42;

mat_menu[21,3]:=6;

mat_menu[22,0]:=2;     mat_menu[22,1]:=0;
mat_menu[22,2]:=26;    mat_menu[22,3]:=46;
mat_menu[22,4]:=23;    mat_menu[22,5]:=25;

mat_menu[23,0]:=2;     mat_menu[23,1]:=0;
mat_menu[23,2]:=27;    mat_menu[23,3]:=46;
mat_menu[23,4]:=24;    mat_menu[23,5]:=22;

mat_menu[24,0]:=2;     mat_menu[24,1]:=0;
mat_menu[24,2]:=28;    mat_menu[24,3]:=46;
```

```
mat_menu[24,4]:=25;   mat_menu[24,5]:=23;

mat_menu[25,0]:=2;    mat_menu[25,1]:=0;
mat_menu[25,2]:=29;   mat_menu[25,3]:=46;
mat_menu[25,4]:=22;   mat_menu[25,5]:=24;

mat_menu[26,0]:=2;    mat_menu[26,1]:=0;
mat_menu[26,2]:=26;   mat_menu[26,3]:=46;
mat_menu[26,4]:=23;   mat_menu[26,5]:=25;

mat_menu[27,0]:=2;    mat_menu[27,1]:=0;
mat_menu[27,2]:=27;   mat_menu[27,3]:=46;
mat_menu[27,4]:=24;   mat_menu[27,5]:=22;

mat_menu[28,0]:=2;    mat_menu[28,1]:=0;
mat_menu[28,2]:=28;   mat_menu[28,3]:=46;
mat_menu[28,4]:=25;   mat_menu[28,5]:=23;

mat_menu[29,0]:=2;    mat_menu[29,1]:=0;
mat_menu[29,2]:=45;   mat_menu[29,3]:=45;
mat_menu[29,4]:=45;   mat_menu[29,5]:=45;

mat_menu[31,0]:=0;    mat_menu[31,1]:=4;
mat_menu[31,2]:=33;   mat_menu[31,3]:=6;
mat_menu[31,4]:=32;   mat_menu[31,5]:=32;

mat_menu[32,0]:=0;    mat_menu[32,1]:=4;
mat_menu[32,2]:=34;   mat_menu[32,3]:=6;
mat_menu[32,4]:=31;   mat_menu[32,5]:=31;

mat_menu[33,0]:=0;    mat_menu[33,1]:=4;
mat_menu[33,2]:=43;   mat_menu[33,3]:=43;
mat_menu[33,4]:=43;   mat_menu[33,5]:=43;

mat_menu[34,0]:=0;    mat_menu[34,1]:=4;
mat_menu[34,2]:=44;   mat_menu[34,3]:=44;
mat_menu[34,4]:=44;   mat_menu[34,5]:=44;

mat_menu[36,0]:=1;    mat_menu[36,1]:=5;
mat_menu[36,2]:=7;    mat_menu[36,3]:=7;
mat_menu[36,4]:=8;    mat_menu[36,5]:=13;

mat_menu[37,0]:=1;    mat_menu[37,1]:=5;
mat_menu[37,2]:=8;    mat_menu[37,3]:=8;
mat_menu[37,4]:=9;    mat_menu[37,5]:=7;

mat_menu[38,0]:=1;    mat_menu[38,1]:=5;
mat_menu[38,2]:=9;    mat_menu[38,3]:=9;
mat_menu[38,4]:=10;   mat_menu[38,5]:=8;

mat_menu[39,0]:=1;    mat_menu[39,1]:=5;
mat_menu[39,2]:=10;   mat_menu[39,3]:=10;
mat_menu[39,4]:=11;   mat_menu[39,5]:=9;
```

```
mat_menu[40,0]:=1;    mat_menu[40,1]:=5;
mat_menu[40,2]:=11;   mat_menu[40,3]:=11;
mat_menu[40,4]:=12;   mat_menu[40,5]:=10;

mat_menu[41,0]:=1;    mat_menu[41,1]:=5;
mat_menu[41,2]:=12;   mat_menu[41,3]:=12;
mat_menu[41,4]:=13;   mat_menu[41,5]:=11;

mat_menu[42,0]:=1;    mat_menu[42,1]:=5;
mat_menu[42,2]:=13;   mat_menu[42,3]:=13;
mat_menu[42,4]:=7;    mat_menu[42,5]:=12;

mat_menu[43,0]:=0;    mat_menu[43,1]:=4;
mat_menu[43,2]:=31;   mat_menu[43,3]:=31;
mat_menu[43,4]:=32;   mat_menu[43,5]:=32;

mat_menu[44,0]:=0;    mat_menu[44,1]:=4;
mat_menu[44,2]:=32;   mat_menu[44,3]:=32;
mat_menu[44,4]:=31;   mat_menu[44,5]:=31;

mat_menu[45,0]:=2;    mat_menu[45,1]:=0;
mat_menu[45,2]:=25;   mat_menu[45,3]:=25;
mat_menu[45,4]:=22;   mat_menu[45,5]:=24;

mat_menu[46,0]:=2;    mat_menu[46,1]:=0;
mat_menu[46,2]:=1;    mat_menu[46,3]:=-4;
mat_menu[46,4]:=-5;   mat_menu[46,5]:=-5;

mat_menu[47,0]:=2;    mat_menu[47,1]:=0;
mat_menu[47,2]:=-5;   mat_menu[47,3]:=-5;
mat_menu[47,4]:=-4;   mat_menu[47,5]:=-4;
```

5.4.5.- Descripción del autómata usado para la detección y corrección de errores cometidos durante la especificación del texto de descripción del dato a nombrar.

La descripción de los autómatas del sistema, está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción de los estados: Esta sección contiene la explicación de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habrá de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección se muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción de los estados del autómata:

Estado inicial:
En el caso de este autómata, el estado inicial no es fijo, sino que varía dependiendo del tipo de error que haya originado la ejecución de este autómata. Por esta razón, el estado inicial se calcula en base al resultado de la rutina

que efectúa el análisis semantico de las palabras del texto.

La función que permite este calculo, es la siguiente:

```
edo_error_actual:=resultado-14;
```

En donde "edo_error_actual" representa al estado inicial del automata.
"resultado" representa al resultado generado por la rutina que efectúa el análisis semantico de las palabras del texto. "resultado" contiene el codigo que identifica al tipo de error identificado en dicha rutina.

A continuación se describen las acciones efectuadas en cada uno de los estados del automata:

Estado 0:
Acciones:
0.1) Explicar que no es posible aceptar el punto final pues según la sintaxis, no se ha terminado la descripción del dato.

Estado 1:
Acciones:
1.1) Explicar que no es posible, dada la sintaxis del lenguaje, aceptar a la preposicion que se escribió.
1.2) Desplegar la preposición escrita por el usuario.

Estado 2:
Acciones:
2.1) Indicar que la palabra que se escribio, no es palabra clase.
2.2) Busca en la lista de palabras clase, alguna o algunas que se parezcan con la que se escribio.
2.3) Si se encuentra una palabra clase semejante, indica que puede cambiarse la palabra escrita por esa palabra clase semejante, o bien que el usuario escriba nuevamente la palabra.
2.4) Si se encuentra más de una palabra clase, forma una lista de palabras semejantes. LLama al automata encargado de la consulta a las listas. Lista consultada: Palabras semejantes.
2.5) Si no se encontró ninguna palabra clase semejante a la que se escribió, entonces llama al autómata encargado de la consulta a las listas. Lista consultada: Palabras clase.
2.6) Estado terminal del autómata.

Estado 3:
Acciones:
3.1) Explicar que no es posible, dada la sintaxis del lenguaje, aceptar al articulo que se escribió.
3.2) Desplegar el artículo escrito por el usuario.

Estado 4:
Acciones:
4.1) Indicar que la palabra que se escribió, no es palabra clase.
4.2) Busca en la lista de palabras clase, alguna o algunas que se parezcan con la que se escribió.
4.3) Si se encuentra una palabra clase semejante, indica que puede cambiarse la palabra escrita por esa palabra clase semejante, o bien que el usuario escriba nuevamente la palabra.
4.4) Si se encuentra más de una palabra clase, forma una lista de palabras semejantes. LLama al autómata encargado de la consulta a las listas. Lista consultada: Palabras semejantes.
4.5) Si no se encontró ninguna palabra clase semejante a la que se escribió, entonces llama al autómata encargado de la consulta a las listas. Lista consultada: Palabras clase.
4.6) Estado terminal del autómata.

Estado 6:
Acciones:
6.1) Indicar que la palabra que se escribió, no es palabra entidad.
6.2) Busca en la lista de palabras entidad, alguna o algunas que se parezcan con la que se escribió.
6.3) Si se encuentra una palabra entidad semejante, indica que puede cambiarse la palabra escrita por esa palabra semejante, o bien que el usuario escriba nuevamente la palabra.
6.4) Si se encuentra más de una palabra entidad, forma una lista de palabras semejantes. LLama al autómata encargado de la consulta a las listas. Lista consultada: Palabras semejantes.
6.5) Si no se encontró ninguna palabra entidad semejante a la que se escribió, entonces llama al autómata encargado de la consulta a las listas. Lista consultada: Palabras entidad.

Estado 7:
Acciones:
7.1) Indicar que la palabra que se escribió, no es palabra entidad.
7.2) Busca en la lista de palabras entidad, alguna o algunas que se parezcan con la que se escribió.
7.3) Si se encuentra una palabra entidad semejante, indica que puede cambiarse la palabra escrita por esa palabra semejante, o bien que el usuario escriba nuevamente la palabra.
7.4) Si se encuentra más de una palabra entidad, forma una lista de palabras semejantes. LLama al autómata encargado de la consulta a las listas. Lista consultada: Palabras semejantes.

7.5) Si no se encontró ninguna palabra entidad semejante a la que se escribio, entonces llama al automata encargado de la consulta a las listas. Lista consultada: Palabras entidad.
7.6) Estado terminal del automata.

Estado 8:
Acciones:
8.1) Indicar que la palabra que se escribió, no es palabra clase.
8.2) Busca en la lista de palabras clase, alguna o algunas que se parezcan con la que se escribió.
8.3) Si se encuentra una palabra clase semejante, indica que puede cambiarse la palabra escrita por esa palabra clase semejante, o bien que el usuario escriba nuevamente la palabra.
8.4) Si se encuentra más de una palabra clase, forma una lista de palabras semejantes. LLama al autómata encargado de la consulta a las listas. Lista consultada: Palabras semejantes.
8.5) Si no se encontró ninguna palabra clase semejante a la que se escribió, entonces llama al autómata encargado de la consulta a las listas. Lista consultada: Palabras clase.
8.6) Estado terminal del autómata.

Estado 13:
Acciones:
13.1) Explicar que dada la sintaxis del lenguaje, no es posible aceptar a la palabra entidad que se escribió.
13.2) Si no es palabra entidad, pero existe semejanza con alguna de ellas, efectuar la acción anterior.

Estado 14:
Acciones:
14.1) Explicar que dada la sintaxis del lenguaje, no es posible aceptar a la palabra clase que se escribió.
14.2) Si no es palabra clase, pero existe semejanza con alguna de ellas, efectuar la acción anterior.

Estado 16:
Acciones:
16.1) Indicar que la palabra escrita no fue preposición.
16.2) Desplegar la palabra que escribió el usuario.

Estado 17:
Acciones:
17.1) Explicar que no es posible aceptar vacíos, debe escribirse algo para ser analizado por el sistema.

Estado 20:
Acciones:
20.1) Mostrar los articulos válidos en el sistema

```
Estado  21:
  Estado terminal del autómata.
  Acciones:
  21.1) Mostrar las palabras clase válidas en el sistema
        Llamar al autómata encargado del manejo de la
        consulta a listas. Lista consultada: Palabras clase.

Estado  22:
  Acciones:
  22.1) Mostrar las preposiciones válidas en el sistema.

Estado  23:
  Acciones:
  23.1) Explicar adjetivos. (ventana 16)

Estado  24:
  Acciones:
  24.1) Explicar sustantivos. (ventana 17)

Estado  25:
  25.1) Mostrar la lista de palabras entidad válidas. Llamar
        al autómata encargado del manejo de la consulta a
        listas. Lista consultada: Palabras entidad.

Estado  26:
  Acciones:
  26.1) Explicar la función del punto final. (ventana 10)
```

Función de transición de estados:

```
if not(fin_error) then
  begin
    err_sigte:=
         mat_error[edo_error_actual,
                   (oprime_esc_int+edo_anterior*2)];
    edo_error_actual:=err_sigte;
  end;
until (edo_error_actual=0) or (fin_error);
```

Matriz de transición de estados del autómata:

```
for ix:=0 to max_col_err do
  for iy:=0 to max_err do
    mat_error[iy,ix]:=0;

        mat_error[ 0, 1]:=20; mat_error[ 1, 1]:=20;
        mat_error[17, 1]:=20; mat_error[20, 1]:=21;
        mat_error[ 0, 3]:=21; mat_error[ 1, 3]:=21;
```

```
mat_error[ 3, 3]:=21; mat_error[17, 3]:=21;
mat_error[ 0, 5]:=22; mat_error[ 1, 5]:=22;
mat_error[ 3, 5]:=22; mat_error[17, 5]:=22;
mat_error[22, 5]:=23;
mat_error[ 0, 7]:=24; mat_error[ 1, 7]:=24;
mat_error[ 3, 7]:=24; mat_error[17, 7]:=24;
mat_error[13, 7]:=24; mat_error[14, 7]:=24;
mat_error[ 0, 9]:=23; mat_error[ 1, 9]:=23;
mat_error[ 3, 9]:=23; mat_error[17, 9]:=23;
mat_error[13, 9]:=23; mat_error[14, 9]:=23;
mat_error[23, 9]:=22;
mat_error[ 0,11]:=22; mat_error[ 3,11]:=22;
mat_error[17,11]:=22;
mat_error[ 0,13]:=20; mat_error[ 1,13]:=20;
mat_error[17,13]:=20; mat_error[20,13]:=25;
mat_error[ 0,15]:=25; mat_error[ 1,15]:=25;
mat_error[ 3,15]:=25; mat_error[17,15]:=25;
mat_error[ 3,17]:=23; mat_error[17,17]:=23;
mat_error[22,17]:=26;
mat_error[23,17]:=22;
mat_error[ 0,19]:=22; mat_error[ 3,19]:=22;
mat_error[17,19]:=22; mat_error[22,19]:=26;
mat_error[ 0,23]:=20; mat_error[ 1,23]:=20;
mat_error[17,23]:=20; mat_error[20,23]:=24;
mat_error[ 0,25]:=24; mat_error[ 1,25]:=24;
mat_error[ 3,25]:=24; mat_error[17,25]:=24;
mat_error[ 3,27]:=26; mat_error[17,27]:=26;
mat_error[23,27]:=22;
mat_error[26,27]:=23;
mat_error[13, 1]:=20; mat_error[13, 3]:=21;
mat_error[13, 5]:=22; mat_error[13, 7]:=24;
mat_error[13, 9]:=23; mat_error[13,11]:=22;
mat_error[14, 5]:=22; mat_error[14, 7]:=24;
mat_error[14, 9]:=23; mat_error[14,11]:=22;
mat_error[14,13]:=20; mat_error[14,15]:=25;
mat_error[14,17]:=23; mat_error[14,19]:=22;
mat_error[14,23]:=20; mat_error[14,25]:=24;
mat_error[14,27]:=26;
mat_error[16,19]:=22; mat_error[16,11]:=22;
```

5.4.6.- Descripción del autómata utilizado para la definición de reglas de abreviación que se usan para la generación de nombres primarios y nombres de programación.

La descripción de los autómatas del sistema, está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción de los estados: Esta sección contiene la explicación de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habrá de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección se muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción de los estados del autómata

Estado inicial : estado 0

A continuación se describen las acciones de cada uno de los estados del autómata:

```
Estado -1:
  Acciones:
 -1.1) Asigna la ventana -12 como ventana actual del sistema.
 -1.2) Desactiva el tercer renglón de esta ventana.
 -1.3) Activa el segundo renglón.
 -1.4) Almacena en el tope de la pila de estados al
       estado -1.
 -1.5) Si se hicieron actualizaciones en las reglas de
       abreviacion, entonces:
       -1.5.1 Muestra al usuario las reglas de abreviación
              con sus valores actualizados.
       -1.5.2 Verifica que el usuario realmente desea que
              estas actualizaciones se almacenen en disco. En
              caso afirmativo entonces se almacenan en disco,
              de lo contrario se desechan.

Estado 0:
  Acciones:
  0.1) Despliega ventana -12
       Reglas para diccionarios de datos
       Reglas para ambientes de programación.
  0.2) Activa renglón "Reglas para diccionarios de datos"
  0.3) Coloca en el tope de la pila de estados al estado -1


Estado 1:
  Acciones:
  1.1) Quita la última ventana desplegada en el monitor
       Con esta acción se quita también al elemento que
       se encuentra en el tope de la pila de estados
  1.2) Si con la acción anterior la pila de estados se
       quedó vacia:
       entonces: 1.2.1) Fin de ejecución del autómata
       si no:    1.2.2) Toma al estado almacenado en el tope
                        de la pila de estados y lo asigna como
                        estado actual del autómata.

Estado 2:
  Acciones:
  2.1) Toma el estado anterior del autómata y lo asigna como
       estado actual.

Estado 3:
  Acciones:
  3.1) Activa el renglón "Reglas de diccionarios de datos"
       de la ventana desplegada en el estado 0.

Estado 4:
  Acciones:
  4.1) Despliega ventana -13
        Selección del diccionario de datos
        # máximo caracteres/palabra_abreviada
        # mínimo caracteres/palabra_abreviada
        # máximo palabras_abreviadas/nombre_primario
```

Orden de palabras_abreviadas en el nombre
Caracteres conectores de palabras abreviadas
Eliminación de letras
Sustituciones
Uso de abreviaciones especiales

4.2) Activa renglón "Selección del diccionario de datos" de esta ventana.
4.3) Coloca en el tope de la pila de estados al estado 13.

Estado 5:
Acciones:
5.1) Activa renglón "# máximo caracteres/palabra_abreviada" de la ventana desplegada en el estado 4

Estado 6:
Acciones:
6.1) Activa renglón "# mínimo caracteres/palabra_abreviada" de la ventana desplegada en el estado 4

Estado 7:
Acciones:
7.1) Activa renglón "# máximo palabras_abreviadas/ nombre_primario" de la ventana desplegada en el estado 4.

Estado 8:
Acciones:
8.1) Activa renglón "Orden de palabras_abreviadas en el nombre" de la ventana desplegada en el estado 4

Estado 9:
Acciones:
9.1) Activa renglón "Caracteres conectores de palabras abreviadas" de la ventana desplegada en el estado 4

Estado 10:
Acciones:
10.1) Activa renglón "Eliminación de letras" de la ventana desplegada en el estado 4

Estado 11:
Acciones:
11.1) Activa renglón "Sustituciones" de la ventana desplegada en el estado 4

Estado 12:
Acciones:
12.1) Activa renglón "Uso de abreviaciones especiales" de la ventana desplegada en el estado 4

Estado 13:
Acciones:
13.1) Activa renglón "Selección de diccionario de datos" de la ventana desplegada en el estado 4

```
Estado 14:
  Acciones:
  14.1) Despliega ventana -14
        Seleccion del ambiente de programación
        # maximo caracteres/palabra_abreviada
        # minimo caracteres/palabra_abreviada
        # máximo palabras_abreviadas/nombre_primario
        Orden de palabras_abreviadas en el nombre
        Caracteres conectores de palabras abreviadas
        Eliminación de letras
        Sustituciones
        Uso de abreviaciones especiales
  14.2) Activa    renglon    "Seleccion    del    ambiente    de
        programacion" de esta ventana.
  14.3) Almacena  en el tope de la pila de estados al
        estado 23.

Estado 15:
  Acciones:
  15.1) Activa renglón "# máximo caracteres/palabra
        abreviada" de la ventana desplegada en el estado 14

Estado 16:
  Acciones:
  16.1) Activa renglón "# mínimo caracteres/palabra
        abreviada" de la ventana desplegada en el estado 14

Estado 17:
  Acciones:
  17.1) Activa renglón "# máximo palabras_abreviadas/nombre
        primario" de la ventana desplegada en el estado 14

Estado 18:
  Acciones:
  18.1) Activa renglón "Orden de palabras_abreviadas en el
        nombre"   de la ventana desplegada en el estado 14

Estado 19:
  Acciones:
  19.1) Activa renglón "Caracteres conectores de palabras
        abreviadas" de la ventana desplegada en el estado 14

Estado 20:
  Acciones:
  20.1) Activa renglón "Eliminación de letras"
        de la ventana desplegada en el estado 14

Estado 21:
  Acciones:
  21.1) Activa renglón "Sustituciones"
        de la ventana desplegada en el estado 14

Estado 22:
```

Acciones:
22.1) Activa renglón "Uso de abreviaciones especiales" de la ventana desplegada en el estado 14

Estado 23:
Acciones:
23.1) Activa renglón "Selección de diccionario de datos" de la ventana desplegada en el estado 14

Estado 24:
Acciones:
24.1) Almacena en el tope de la pila de estados al estado 13.
24.2) Si las reglas de abreviación sufrieron alguna actualización, entonces:
24.2.1) Muestra al usuario las reglas de abreviación actualizadas.
24.2.2) Verifica que realmente el usuario desea que estas reglas de abreviación se almacenen en disco. En caso afirmativo, se almacenan; de lo contrario se desechan los cambios.
24.3) Llama a la rutina encargada de manejar las consultas a las listas del sistema. Lista consultada: Nombres de diccionarios de datos.
24.4) Recibe el nombre del diccionario de datos cuyas reglas de abreviación se desea actualizar o consultar.
24.5) Consulta que el nombre especificado exista en el acervo de nombres de diccionarios de datos.
24.6) Si el nombre de diccionario de datos existe en el acervo del sistema, entonces se cargan sus reglas de abreviación del disco a memoria principal.
24.7) Si el nombre no existe, entonces se verifica que el usuario desea darlo de alta.
24.8) Esta acción se lleva a cabo en el caso de que el usuario esté autorizado:
En caso de que el usuario realmente desea dar de alta al nombre especificado, entonces éste se da de alta y se generan reglas de abreviación estándar para el nuevo nombre. Se actualizan las estructuras de datos dinámicas y archivos del sistema. En caso negativo, se omiten la actualización.

Estado 25:
Acciones:
25.1) Almacena en el tope de la pila de estados al estado 5
25.2) Despliega ventana -15
Valor:
25.3) Llama a rutina que analiza el teclado para leer el numero que especifique el usuario.
25.4) Si el usuario está autorizado a hacer

actualizaciones, entonces actualiza el valor de la regla "maximo número de letras por palabra abreviada".
Indica que la regla se actualizo.

Estado 26:
Acciones:
26.1) Almacena en el tope de la pila de estados al estado 6
26.2) Despliega ventana -15
Valor:
26.3) Llama a la rutina que analiza el teclado para leer el número que especifique el usuario.
26.4) Si el usuario está autorizado a hacer actualizaciones, entonces actualiza el valor de la regla "minimo número de letras por palabra abreviada". Indica que la regla se actualizó.

Estado 27:
Acciones:
27.1) Almacena en el tope de la pila de estados al estado 7.
27.2) Despliege ventana -15
Valor:
27.3) Llama a la rutina que analiza el teclado para leer el número que especifique el usuario.
27.4) Si el usuario está autorizado a hacer actualizaciones, entonces actualiza el valor de "máximo número de palabras por nombre abreviado". Indica que la regla se actualizó.

Estado 28:
Acciones:
28.1) Almacena en el tope de la pila de estados al estado 8
28.2) Despliega ventana -16
Orden de palabras_abreviadas en el nombre
Clase, Entidad y Modificadores
Clase, 1er Modificador, Entidad y Modificadores
Entidad, Clase y Modificadores
Entidad, 1er Modificador, Clase y Modificadores
1er Modificador, Clase, Entidad y Modificadores
1er Modificador, Entidad, Clase y Modificadores
Mismo que en la descripción del dato
Orden seleccionado:
28.3) Activa renglón "Clase, entidad y modificadores" de esta ventana.
28.4) Almacena en el tope de la pila de estados al estado 50
28.5) Despliega el valor de la regla "Orden de palabras en el nombre"

Estado 29:
Acciones:
29.1) Almacena en el tope de la pila de estados al estado 9
29.2) Despliega la ventana -17
Guión -
Subguión _
Punto .
Definir caracter
Selección:

29.3) Activa renglón "Guión - "
de esta ventana.
29.4) Almacena en el tope de la pila de estados al estado 102.
29.5) Despliega el valor de la regla "Caracter conector"

Estado 30:
Acciones:
30.1) Almacena en el tope de la pila de estados al estado 10.
30.2) Despliega la ventana -18
Eliminación de la letra inicial
Eliminación de la letra final
Eliminación de las vocales intermedias
Eliminación de consonantes adyacentes iguales
Eliminación de caracteres numéricos
30.3) Activa renglón "Eliminación de la letra inicial" de esta ventana.
30.4) Almacena en el tope de la pila de estados al estado 42.

Estado 31:
Acciones:
31.1) Almacena en el tope de la pila de estados al estado 11 .
31.2) Despliega ventana -27
Sustitución del gramema de número
Sustitución del gramema de género
Sustitución del caracter "ñ"
31.3) Activa renglón "Sustitución del gramema de número" de esta ventana.
31.4) Almacena en el tope de la pila de estados al estado 47

Estado 32:
Acciones:
32.1) Almacena en el tope de la pila de estados al estado 12
32.2) Despliega ventana -32
Regla de abreviación activa
Regla de abreviación inactiva
Opción seleccionada:
32.3) Activa renglón "Regla de abreviación activa"

de esta ventana.
32.4) Almacena en el tope de la pila de estados al estado 110
32.5) Despliega valor actual de la regla "Abreviación especial"

Estado 33:
Acciones:
33.1) Almacena en el tope de la pila de estados al estado 23.
33.2) Si las reglas de abreviación sufrieron alguna actualización, entonces:
33.2.1) Muestra al usuario las reglas de abreviación actualizadas.
33.2.2) Verifica que realmente el usuario desea que estas reglas de abreviación se almacenen en disco. En caso afirmativo, se almacenan; de lo contrario se desechan los cambios.
33.3) Llama a la rutina encargada de manejar las consultas a las listas del sistema. Lista consultada: Nombres de ambientes de programación.
33.4) Recibe el nombre del ambiente de programación cuyas reglas de abreviación se desea actualizar o consultar.
33.5) Consulta que el nombre especificado exista en el acervo de nombres de ambientes de programación.
33.6) Si el nombre de ambiente de programación existe en el acervo del sistema, entonces se cargan sus reglas de abreviación del disco a memoria principal.
33.7) Si el nombre no existe, entonces se verifica que el usuario desea darlo de alta.
33.8) Esta acción se lleva a cabo en el caso de que el usuario esté autorizado:
En caso de que el usuario realmente desea dar de alta al nombre especificado, entonces éste se da de alta y se generan reglas de abreviación estándar para el nuevo nombre. Se actualizan las estructuras de datos dinámicas y archivos del sistema. En caso negativo, se omiten la actualizacion.

Estado 34:
Acciones:
34.1) Almacena en el tope de la pila de estados al estado 15
34.2) Despliega ventana -15
Valor:
34.3) Llama a rutina que analiza el teclado para leer el numero que especifique el usuario.
34.4) Si el usuario está autorizado a hacer actualizaciones, entonces actualiza el valor de la regla "máximo número de letras por palabra abreviada".
Indica que la regla se actualizó.

Estado 35:
Acciones:
35.1) Almacena en el tope de la pila de estados al estado 16
35.2) Despliega ventana -15
Valor:
35.3) Llama a rutina que analiza el teclado para leer el número que especifique el usuario.
35.4) Si el usuario está autorizado a hacer actualizaciones, entonces actualiza el valor de la regla "mínimo número de letras por palabra abreviada". Indica que la regla se actualizó.

Estado 36:
Acciones:
36.1) Almacena en el tope de la pila de estados al estado 17
36.2) Despliega ventana -15
Valor:
36.3) Llama a rutina que analiza el teclado para leer el numero que especifique el usuario.
36.4) Si el usuario está autorizado a hacer actualizaciones, entonces actualiza el valor de la regla "máximo número de palabras por nombre". Indica que la regla se actualizò.

Estado 37:
Acciones:
37.1) Almacena en el tope de la pila de estados al estado 18.
37.2) Despliega ventana -16
Orden de palabras abreviadas en el nombre
Clase, Entidad y Modificadores
Clase, 1er Modificador, Entidad y Modificadores
Entidad, Clase y Modificadores
Entidad, 1er Modificador, Clase y Modificadores
1er Modificador, Clase, Entidad y Modificadores
1er Modificador, Entidad, Clase y Modificadores
Mismo que en la descripción del dato
Orden seleccionado:
37.3) Activa renglón "Clase, entidad y modificadores" de esta ventana.
37.4) Almacena en el tope de la pila de estados al estado 50 .
37.5) Despliega el valor de la regla "Orden de palabras en el nombre"

Estado 38:
Acciones:
38.1) Almacena en el tope de la pila de estados al estado 19 .
38.2) Despliega la ventana -17
Guión -
Subguión _

Punto .
Definir caracter
Selección:

38.3) Activa renglón "Guión - "
de esta ventana.
38.4) Almacena en el tope de la pila de estados al
estado 102 .
38.5) Despliega el valor de la regla
"Caracter conector"

Estado 39:
Acciones:
39.1) Almacena en el tope de la pila de estados
al estado 20
39.2) Despliega la ventana -18
Eliminación de la letra inicial
Eliminación de la letra final
Eliminación de las vocales intermedias
Eliminación de consonantes adyacentes iguales
Eliminación de caracteres numéricos
39.3) Activa renglón "Eliminación de la letra inicial"
de esta ventana.
39.4) Almacena en el tope de la pila de estados al
estado 42.

Estado 40:
Acciones:
40.1) Almacena en el tope de la pila de estados al
estado 21
40.2) Despliega ventana -27
Sustitución del gramema de número
Sustitución del gramema de género
Sustitución del caracter "ñ"
40.3) Activa renglón "Sustitución del gramema de número"
de esta ventana.
40.4) Almacena en el tope de la pila de estados al
estado 47

Estado 41:
Acciones:
41.1) Almacena en el tope de la pila de estados
al estado 22.
41.2) Despliega ventana -32
Regla de abreviación activa
Regla de abreviación inactiva
Opción seleccionada:
41.3) Activa renglón "Regla de abreviación activa"
de esta ventana.
41.4) Almacena en el tope de la pila de estados
al estado 110.
41.4) Despliega valor actual de la regla
"Abreviación especial".

Estado 42:
Acciones:
42.1) Activa renglón "Eliminación de la letra inicial" de la ventana desplegada en el estado 38.

Estado 43:
Acciones:
43.1) Activa renglón "Eliminación de la letra final" de la ventana desplegada en el estado 38.

Estado 44:
Acciones:
44.1) Activa renglón "Eliminación de las vocales intermedias" de la ventana desplegada en el estado 38

Estado 45:
Acciones:
45.1) Activa renglón "Eliminación de consonantes adyacentes iguales" de la ventana desplegada en el estado 38.

Estado 46:
Acciones:
46.1) Activa renglón "Eliminación de caracteres numéricos" de la ventana desplegada en el estado 38.

Estado 47:
Acciones:
47.1) Activa renglón "Sustitución del gramema de número" de la ventana desplegada en el estado 40.

Estado 48:
Acciones:
48.1) Activa renglón "Sustitución del gramema de género" de la ventana desplegada en el estado 40.

Estado 49:
Acciones:
49.1) Activa renglón 'Sustitución del caracter "ñ"' de la ventana desplegada en el estado 40.

Estado 50:
Acciones:
50.1) Activa renglón "Clase, Entidad y Modificadores" de la ventana desplegada en el estado 37.

Estado 51:
Acciones:
51.1) Activa renglón "Clase, 1er Modificador, Entidad y Modificadores" de la ventana desplegada en el estado 37.

Estado 52:
Acciones:

52.1) Activa renglón "Entidad, Clase y Modificadores" de la ventana desplegada en el estado 37.

Estado 53:
Acciones:
53.1) Activa renglón "Entidad, 1er Modificador, Clase y Modificadores" de la ventana desplegada en el estado 37.

Estado 54:
Acciones:
54.1) Activa renglón "1er Modificador, Clase, Entidad y Modificadores" de la ventana desplegada en el estado 37.

Estado 55:
Acciones:
55.1) Activa renglón "1er Modificador, Entidad, Clase y Modificadores" de la ventana desplegada en el estado 37.

Estado 56:
Acciones:
56.1) Activa renglón "Mismo que en la descripción del dato" de la ventana desplegada en el estado 37.

Estado 57:
Acciones:
57.1) Almacena en el tope de la pila de estados al estado 47.
57.2) Despliega ventana -30
Sustitución de singular a plural
Sustitución de plural a singular
No hay sustitución del gramema de número
Opción seleccionada:
57.3) Activa renglón "Sustitución de singular a plural" de esta ventana.
57.4) Almacena en el tope de la pila de estados al estado 59.
57.5) Despliega el valor de la regla "Sustitución del gramema de número".

Estado 58:
Acciones:
58.1) Almacena en el tope de la pila de estados al estado 48.
58.2) Despliega ventana -31
Sustitución de masculino a femenino
Sustitución de femenino a masculino
No hay sustitución del gramema de género
Opción seleccionada:
58.3) Activa renglón "Sustitución de masculino a femenino" de esta ventana.
58.4) Almacena en el tope de la pila de estados

al estado 62.
58.5) Despliega el valor de la regla "Sustitución del gramema de género"

Estado 59:
Acciones:
59.1) Activa renglón "Sustitución de plural a singular" de la ventana desplegada en el estado 57.

Estado 60:
Acciones:
60.1) Activa renglón "Sustitución de singular a plural" de la ventana desplegada en el estado 57.

Estado 61:
Acciones:
61.1) Activa renglón "No hay sustitución del gramema de número" de la ventana desplegada en el estado 57.

Estado 62:
Acciones:
62.1) Activa renglón "Sustitución de masculino a femenino" de la ventana desplegada en el estado 58.

Estado 63:
Acciones:
63.1) Activa renglón "Sustitución de femenino a masculino" de la ventana desplegada en el estado 58.

Estado 64:
Acciones:
64.1) Activa renglón "No hay sustitución del gramema de género" de la ventana desplegada en el estado 58.

Estado 65:
Acciones: (Sólo si el usuario está autorizado)
65.1) Asigna el valor 'Clase, Entidad y Modificadores' a la regla "Orden de las palabras en el nombre"
65.2) Indica que se cambió la regla.
65.3) Despliega el valor de esta regla en el monitor.

Estado 66:
Acciones: (Sólo si el usuario está autorizado)
66.1) Asigna el valor 'ler Modificador, Entidad y Modificadores' a la regla "Orden de las palabras en el nombre"
66.2) Indica que se cambió la regla.
66.3) Despliega el valor de esta regla en el monitor.

Estado 67:
Acciones: (Sólo si el usuario está autorizado)
67.1) Asigna el valor 'Entidad, Clase y Modificadores' a la regla "Orden de las palabras en el nombre"
67.2) Indica que se cambió la regla.

67.3) Despliega el valor de esta regla en el monitor.

Estado 68:
Acciones: (Sólo si el usuario está autorizado)
68.1) Asigna el valor 'Entidad, 1er Modificador, Clase y Modificadores'a la regla "Orden de las palabras en el nombre".
68.2) Indica que se cambió la regla.
68.3) Despliega el valor de esta regla en el monitor.

Estado 69:
Acciones: (Sólo si el usuario está autorizado)
69.1) Asigna el valor '1er Modificador, Clase, Entidad y Modificadores'a la regla "Orden de las palabras en el nombre" .

69.2) Indica que se cambió la regla.
69.3) Despliega el valor de esta regla en el monitor.

Estado 70:
Acciones: (Sólo si el usuario está autorizado)
70.1) Asigna el valor '1er Modificador, Entidad, Clase y Modificadores'a la regla "Orden de las palabras en el nombre".
70.2) Indica que se cambió la regla.
70.3) Despliega el valor de esta regla en el monitor.

Estado 71:
Acciones: (Sólo si el usuario está autorizado)
71.1) Asigna el valor 'Mismo que en la descripción del dato' a la regla "Orden de las palabras en el nombre".
71.2) Indica que se cambió la regla.
71.3) Despliega el valor de esta regla en el monitor.

Estado 72:
Acciones: (Sólo si el usuario está autorizado)
72.1) Asigna el valor 'Sustitución de singular a plural' a la regla "Sustitución del gramema de número".
72.2) Indica que se cambió la regla.
72.3) Despliega el valor de esta regla en el monitor.

Estado 73:
Acciones: (Sólo si el usuario está autorizado)
73.1) Asigna el valor 'Sustitución de plural a singular' a la regla "Sustitución del gramema de número".
73.2) Indica que se cambió la regla.
73.3) Despliega el valor de esta regla en el monitor.

Estado 74:
Acciones: (Sólo si el usuario está autorizado)
74.1) Asigna el valor 'No hay sustitución' a la regla "Sustitución del gramema de número"
74.2) Indica que se cambió la regla.

74.3) Despliega el valor de esta regla en el monitor.

Estado 75:
Acciones: (Sólo si el usuario está autorizado)
75.1) Asigna el valor 'Sustitución de masculino a femenino" a la regla "Sustitución del gramema de género".
75.2) Indica que se cambió la regla.
75.3) Despliega el valor de esta regla en el monitor.

Estado 76:
Acciones: (Sólo si el usuario está autorizado)
76.1) Asigna el valor 'Sustitución de femenino a masculino" a la regla "Sustitución del gramema de género".
76.2) Indica que se cambió la regla.
76.3) Despliega el valor de esta regla en el monitor.

Estado 76:
Acciones: (Sólo si el usuario está autorizado)
76.1) Asigna el valor 'No hay sustitución' a la regla "Sustitución del gramema de género".
76.2) Indica que se cambió la regla.
76.3) Despliega el valor de esta regla en el monitor.

Estado 77:
Acciones:
77.1) Almacena en el tope de la pila de estados al estado 42.
77.2) Despliega ventana -32
Regla de abreviación activa
Regla de abreviación inactiva
Opción seleccionada:
77.3) Activa renglón"Regla de abreviación activa" de esta ventana.
77.4) Almacena en el tope de la pila de estados al estado 82.
77.5) Despliega el valor de la regla "Eliminación de la letra inicial".

Estado 78:
Acciones:
78.1) Almacena en el tope de la pila de estados al estado 43.
78.2) Despliega ventana -32
Regla de abreviación activa
Regla de abreviación inactiva
Opción seleccionada:
78.3) Activa renglón"Regla de abreviación activa" de esta ventana.
78.4) Almacena en el tope de la pila de estados al estado 84.
78.5) Despliega el valor de la regla "Eliminación de la letra final"

Estado 79:
  Acciones:
  79.1) Almacena en el tope de la pila de estados
        al estado 44.
  79.2) Despliega ventana -32
         Regla de abreviación activa
         Regla de abreviacion inactiva
         Opción seleccionada:
  79.3) Activa renglón"Regla de abreviacion activa"
        de esta ventana.
  79.4) Almacena en el tope de la pila de estados
        al estado 86.
  79.5) Despliega el valor de la regla
        "Eliminación de vocales intermedias"

Estado 80:
  Acciones:
  80.1) Almacena en el tope de la pila de estados
        al estado 45.
  80.2) Despliega ventana -32
         Regla de abreviación activa
         Regla de abreviación inactiva
         Opción seleccionada:
  80.3) Activa renglón"Regla de abreviación activa"
        de esta ventana.
  80.4) Almacena en el tope de la pila de estados
        al estado 88.
  80.5) Despliega el valor de la regla
        "Eliminación de consonantes adyacentes"

Estado 81:
  Acciones:
  81.1) Almacena en el tope de la pila de estados
        al estado 46.
  81.2) Despliega ventana -32
         Regla de abreviación activa
         Regla de abreviación inactiva
         Opción seleccionada:
  81.3) Activa renglón"Regla de abreviación activa"
        de esta ventana.
  81.4) Almacena en el tope de la pila de estados
        al estado 90.
  81.5) Despliega el valor de la regla
        "Eliminación de caracteres numéricos"

Estado 82:
  Acciones:
  82.1) Activa renglón "Regla de abreviación activa"
        de la ventana desplegada en el estado

Estado 83:
  Acciones:
  83.1) Activa renglón "Regla de abreviación inactiva"
        de la ventana desplegada en el estado

Estado 84:
Acciones:
84.1) Activa renglón "Regla de abreviación activa"
de la ventana desplegada en el estado

Estado 85:
Acciones:
85.1) Activa renglón "Regla de abreviación inactiva"
de la ventana desplegada en el estado

Estado 86:
Acciones:
86.1) Activa renglón "Regla de abreviación activa"
de la ventana desplegada en el estado

Estado 87:
Acciones:
87.1) Activa renglón "Regla de abreviación inactiva"
de la ventana desplegada en el estado

Estado 88:
Acciones:
88.1) Activa renglón "Regla de abreviación activa"
de la ventana desplegada en el estado

Estado 89:
Acciones:
89.1) Activa renglón "Regla de abreviación inactiva"
de la ventana desplegada en el estado

Estado 90:
Acciones:
90.1) Activa renglón "Regla de abreviación activa"
de la ventana desplegada en el estado

Estado 91:
Acciones:
91.1) Activa renglón "Regla de abreviación inactiva"
de la ventana desplegada en el estado

Estado 92:
Acciones: (Sólo si el usuario está autorizado)
92.1) Asigna el valor 'Regla de abreviación activa' a la
regla "Eliminación de la letra inicial"
92.2) Indica que se cambió la regla.
92.3) Despliega el valor de esta regla.

Estado 93:
Acciones: (Sólo si el usuario está autorizado)
93.1) Asigna el valor 'Regla de abreviación inactiva' a la
regla "Eliminación de la letra inicial"
93.2) Indica que se cambió la regla.
93.3) Despliega el valor de esta regla.

Estado 94:
Acciones: (Sólo si el usuario esta autorizado)
94.1) Asigna el valor 'Regla de abreviación activa' a la regla "Eliminación de la letra final"
94.2) Indica que se cambio la regla.
94.3) Despliega el valor de esta regla.

Estado 95:
Acciones: (Solo si el usuario esta autorizado)
95.1) Asigna el valor 'Regla de abreviación inactiva' a la regla "Eliminación de la letra final"
95.2) Indica que se cambió la regla.
95.3) Despliega el valor de esta regla.

Estado 96:
Acciones: (Sólo si el usuario está autorizado)
96.1) Asigna el valor 'Regla de abreviación activa' a la regla "Eliminación de vocales intermedias"
96.2) Indica que se cambió la regla.
96.3) Despliega el valor de esta regla.

Estado 97:
Acciones: (Solo si el usuario esta autorizado)
97.1) Asigna el valor 'Regla de abreviación inactiva' a la regla "Eliminación de vocales intermedias"
97.2) Indica que se cambió la regla.
97.3) Despliega el valor de esta regla.

Estado 98:
Acciones: (Sólo si el usuario está autorizado)
98.1) Asigna el valor 'Regla de abreviación activa' a la regla "Eliminación de consonantes adyacentes"
98.2) Indica que se cambió la regla.
98.3) Despliega el valor de esta regla.

Estado 99:
Acciones: (Sólo si el usuario está autorizado)
99.1) Asigna el valor 'Regla de abreviación inactiva' a la regla "Eliminación de consonantes adyacentes"
99.2) Indica que se cambió la regla.
99.3) Despliega el valor de esta regla.

Estado 100:
Acciones: (Sólo si el usuario está autorizado)
100.1) Asigna el valor 'Regla de abreviación activa' a la regla "Eliminación de caracteres numéricos"
100.2) Indica que se cambió la regla.
100.3) Despliega el valor de esta regla.

Estado 101:
Acciones: (Sólo si el usuario está autorizado)
101.1) Asigna el valor 'Regla de abreviación inactiva' a la regla "Eliminación de caracteres numéricos"

101.2) Indica que se cambió la regla.
101.3) Despliega el valor de esta regla.

Estado 102:
Acciones:
102.1) Activa renglón "Guión -"
de la ventana desplegada en el estado 38.

Estado 103:
Acciones:
103.1) Activa renglón "Subguión _"
de la ventana desplegada en el estado 38.

Estado 104:
Acciones:
104.1) Activa renglón "Punto ."
de la ventana desplegada en el estado 38.

Estado 105:
Acciones:
105.1) Activa renglòn "Definir caracter"
de la ventana desplegada en el estado 38.

Estado 106:
Acciones: (Sólo si el usuario está autorizado)
106.1) Asigna el valor 'Guión' a la regla "Caracter conector entre abreviaciones".
106.2) Indica que se cambió la regla.
106.3) Despliega el valor de esta regla.

Estado 107:
Acciones: (Sólo si el usuario está autorizado)
107.1) Asigna el valor 'Subguión' a la regla "Caracter conector entre abreviaciones".
107.2) Indica que se cambió la regla.
107.3) Despliega el valor de esta regla.

Estado 108:
Acciones: (Sólo si el usuario está autorizado)
108.1) Asigna el valor 'Punto' a la regla "Caracter conector entre abreviaciones".
108.2) Indica que se cambió la regla.
108.3) Despliega el valor de esta regla.

Estado 109:
Acciones:
109.1) Recibe el caracter especial que servirá de unión entre palabras abreviadas de los nombres generados por el sistema.
109.2) Si el usuario oprimió la tecla 'ESC' se pone fin a las acciones de este estado.
109.3) Si el usuario especificó un caracter especial y además cuenta con la autorización para actualizar reglas de abreviación, entonces:

109.3.1) Asigna como caracter de unión entre palabras abreviadas al caracter especificado por el usuario.
109.3.2) Indica que se cambio la regla de abreviación.

Estado 110:
Acciones:
110.1) Activa renglón "Regla de abreviación activa" de la ventana desplegada en el estado

Estado 111:
Acciones:
111.1) Activa renglón "Regla de abreviacion inactiva" de la ventana desplegada en el estado

Estado 112:
Acciones: (Sólo si el usuario está autorizado)
112.1) Asigna el valor 'Regla de abreviación activa' a la regla "Uso de abreviaciones especiales"
112.2) Indica que se cambió la regla.
112.3) Despliega el valor de esta regla.

Estado 113:
Acciones: (Sólo si el usuario está autorizado)
113.1) Asigna el valor 'Regla de abreviación inactiva' a la regla "Uso de abreviaciones especiales"
113.2) Indica que se cambió la regla.
113.3) Despliega el valor de esta regla.

Estado 114:
Acciones:
114.1) Almacena en el tope de la pila de estados al estado 49.
114.2) Recibe el caracter especial que servira para sustituir a la letra 'ñ' en las palabras abreviadas de los nombres generados por el sistema.
114.3) Si el usuario oprimió la tecla 'ESC' se pone fin a las acciones de este estado.
114.4) Si el usuario especificó un caracter especial y además cuenta con la autorización para actualizar reglas de abreviación, entonces:
114.4.1) Asigna como caracter de sustitución de la letra 'ñ' en las palabras abreviadas al caracter especificado por el usuario.
114.4.2) Indica que se cambió la regla de abreviación.

Función de transición de estados utilizada por este autómata:

```
if not(fin_reglas) then
begin
  if not(brinca) then
    begin
      reglas_ant_edo:=reglas_edo_actual;
      if     ((reglas_edo_actual<24)and
              (reglas_edo_actual<>4)and
              (reglas_edo_actual<>14))
         or ((reglas_edo_actual>=28)and
              (reglas_edo_actual<=32))
         or ((reglas_edo_actual>=37)and
              (reglas_edo_actual<=64))
         or ((reglas_edo_actual>=77)and
              (reglas_edo_actual<=91))
         or ((reglas_edo_actual>=102)and
              (reglas_edo_actual<=105))
         or (reglas_edo_actual=110)
         or (reglas_edo_actual=111)

         then reglas_sigte_edo:=
              mat_reglas[reglas_edo_actual,mira_tecla]
         else reglas_sigte_edo:=
              mat_reglas[reglas_edo_actual,2];

      reglas_edo_actual:=reglas_sigte_edo;
    end;
end;
until (reglas_edo_actual=-2) or (fin_reglas);
```

Matriz de transición de estados del autómata:

```
for iy:=-1 to max_ren_reglas do
  for ix:=2 to max_col_reglas do
    mat_reglas[iy,ix]:=-2;
for iy:=0 to max_ren_reglas do
  begin
    mat_reglas[iy,3]:=1;
  end;

mat_reglas[-1,2]:=4;
mat_reglas[-1,4]:=3;
mat_reglas[-1,5]:=3;

mat_reglas[0,2]:=4;
mat_reglas[0,4]:=3;
mat_reglas[0,5]:=3;
```

```
mat_reglas[3,2]:=14;
mat_reglas[3,4]:=-1;
mat_reglas[3,5]:=-1;

for iy:=4 to 12 do
  begin
    mat_reglas[iy,4]:=iy+1;
    mat_reglas[iy,5]:=iy-1;
  end;
mat_reglas[4,5]:=12;

mat_reglas[5,5]:=13;

mat_reglas[13,4]:=5;
mat_reglas[13,5]:=12;

for iy:=14 to 22 do
  begin
    mat_reglas[iy,4]:=iy+1;
    mat_reglas[iy,5]:=iy-1;
  end;
mat_reglas[14,5]:=22;

mat_reglas[15,5]:=23;

mat_reglas[23,4]:=15;
mat_reglas[23,5]:=22;

for iy:=4 to 12 do
  mat_reglas[iy,2]:=iy+20;
mat_reglas[13,2]:=24;

for iy:=14 to 23 do
  mat_reglas[iy,2]:=iy+19;
mat_reglas[23,2]:=33;

for iy:=24 to max_ren_reglas do
  for ix:=2 to max_col_reglas do
    mat_reglas[iy,ix]:=1;
mat_reglas[28,2]:=65;
mat_reglas[28,4]:=51;
mat_reglas[28,5]:=56;
mat_reglas[29,2]:=106;
mat_reglas[29,4]:=103;
mat_reglas[29,5]:=105;
mat_reglas[38,2]:=106;
mat_reglas[38,4]:=103;
mat_reglas[38,5]:=105;
mat_reglas[37,2]:=65;
mat_reglas[37,4]:=51;
mat_reglas[37,5]:=56;
mat_reglas[39,4]:=43;
mat_reglas[30,4]:=43;
mat_reglas[30,5]:=42;
```

```
mat_reglas[39,5]:=42;

for iy:=42 to 45 do
  mat_reglas[iy,4]:=iy+1;
  mat_reglas[46,4]:=42;
for iy:=43 to 46 do
  mat_reglas[iy,5]:=iy-1;
  mat_reglas[42,5]:=46;

mat_reglas[30,2]:=77;
mat_reglas[39,2]:=77;
mat_reglas[42,2]:=77;
mat_reglas[43,2]:=78;
mat_reglas[44,2]:=79;
mat_reglas[45,2]:=80;
mat_reglas[46,2]:=81;

mat_reglas[41,2]:=112;
mat_reglas[41,4]:=111;
mat_reglas[41,5]:=111;
mat_reglas[32,2]:=112;
mat_reglas[32,4]:=111;
mat_reglas[32,5]:=111;

mat_reglas[31,2]:=57;
mat_reglas[40,2]:=57;
mat_reglas[31,4]:=48;
mat_reglas[40,4]:=48;
mat_reglas[31,5]:=49;
mat_reglas[40,5]:=49;
mat_reglas[47,2]:=57;
mat_reglas[47,4]:=48;
mat_reglas[48,2]:=58;
mat_reglas[48,4]:=49;
mat_reglas[49,2]:=114;
mat_reglas[49,4]:=47;
mat_reglas[49,5]:=48;
mat_reglas[48,5]:=47;
mat_reglas[47,5]:=49;

for iy:=50 to 55 do
  mat_reglas[iy,4]:=iy+1;
  mat_reglas[56,4]:=50;
for iy:=51 to 56 do
  mat_reglas[iy,5]:=iy-1;
  mat_reglas[50,5]:=56;

mat_reglas[57,2]:=72;
mat_reglas[57,4]:=60;
mat_reglas[57,5]:=61;
mat_reglas[58,2]:=75;
mat_reglas[58,4]:=63;
mat_reglas[58,5]:=64;
```

```
mat_reglas[59,4]:=60;
mat_reglas[60,4]:=61;
mat_reglas[61,4]:=59;
mat_reglas[59,5]:=61;
mat_reglas[60,5]:=59;
mat_reglas[61,5]:=60;

mat_reglas[62,4]:=63;
mat_reglas[63,4]:=64;
mat_reglas[64,4]:=62;
mat_reglas[62,5]:=64;
mat_reglas[63,5]:=62;
mat_reglas[64,5]:=63;

for iy:=50 to 56 do
  mat_reglas[iy,2]:=iy+15;

for iy:=59 to 64 do
  mat_reglas[iy,2]:=iy+13;

mat_reglas[77,2]:=92;
mat_reglas[78,2]:=94;
mat_reglas[79,2]:=96;
mat_reglas[80,2]:=98;
mat_reglas[81,2]:=100;

for iy:=82 to 91 do
  mat_reglas[iy,2]:=iy+10;

mat_reglas[77,4]:=83;
mat_reglas[78,4]:=85;
mat_reglas[79,4]:=87;
mat_reglas[80,4]:=89;
mat_reglas[81,4]:=91;
mat_reglas[82,4]:=83;
mat_reglas[83,4]:=82;
mat_reglas[84,4]:=85;
mat_reglas[85,4]:=84;
mat_reglas[86,4]:=87;
mat_reglas[87,4]:=86;
mat_reglas[88,4]:=89;
mat_reglas[89,4]:=88;
mat_reglas[90,4]:=91;
mat_reglas[91,4]:=90;

for iy:=77 to 91 do
  mat_reglas[iy,5]:=mat_reglas[iy,4];

for iy:=102 to 105 do
  mat_reglas[iy,2]:=iy+4;
for iy:=102 to 104 do
  mat_reglas[iy,4]:=iy+1;
for iy:=103 to 105 do
  mat_reglas[iy,5]:=iy-1;
```

```
mat_reglas[105,4]:=102;
mat_reglas[102,5]:=105;

mat_reglas[110,2]:=112;
mat_reglas[110,4]:=111;
mat_reglas[110,5]:=111;
mat_reglas[111,2]:=113;
mat_reglas[111,4]:=110;
mat_reglas[111,5]:=110;
```

5.4.7.- Descripción del autómata usado para controlar las ventanas que se usan en la fase de generación de nombres primarios y nombres de programación.

La descripción de los autómatas del sistema, está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción de los estados: Esta sección contiene la explicación de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habrá de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección se muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción de los estados del autómata:

Estado inicial: 0

A continuación se describen las acciones de cada uno de los estados de este autómata:

Estado 0:
  Acciones:

0.1) Llama a la rutina encargada del manejo de consultas a las listas del sistema. Lista consultada: Nombres de diccionarios de datos.
0.2) Recibe del teclado el nombre del diccionario de datos para el que se quiere generar el nombre primario del dato.
0.3) Llama a la rutina que se encarga de buscar en el archivo de reglas de abreviacion, a las reglas del diccionario de datos especificado en el punto anterior.
0.4) Llama a la rutina que se encarga de seleccionar a las palabras claves de la descripción del dato.
0.5) Llama a la rutina que se encarga, en base a las reglas de abreviación del punto 0.3 , de generar al nombre primario.
0.6) Despliega la ventana en la que se muestra el nombre primario generado.
0.7) Si el usuario especificó la generación de reportes, llama a la rutina que se encarga de almacenar en el buffer de generación de documentos, el nombre primario, descripción del dato y diccionario de datos, para que de esta forma se almacene dicha información en disco duro.
0.8) Quita todas las ventanas desplegadas en el monitor.
0.9) Activa la frase "Genera nombres" en la pantalla.
0.10) Despliega la ventana -36
Nombres para diccionarios de datos.
Nombres para ambientes de programación.
0.11) Activa el renglón "Nombres para diccionarios de datos" de esta ventana.

Estado 1:
Acciones:

1.1) Llama a la rutina encargada del manejo de consultas a las listas del sistema. Lista consultada: Nombres de ambientes de programación.
1.2) Recibe del teclado el nombre del ambiente de programación para el que se quiere generar el nombre de programación del dato.
1.3) Llama a la rutina que se encarga de buscar en el archivo de reglas de abreviación, a las reglas del ambiente de programación especificado en el punto anterior.
1.4) Llama a la rutina que se encarga de seleccionar a las palabras claves de la descripción del dato.
1.5) Llama a la rutina que se encarga, en base a las reglas de abreviación del punto 0.3 , de generar al nombre de programación.
1.6) Despliega la ventana en la que se muestra el nombre de programación generado.
1.7) Si el usuario especificó la generación de reportes, llama a la rutina que se encarga de almacenar en el buffer de generación de documentos, el nombre de

programación, descripción del dato y ambiente de programación en el que se usará dicho dato, para que de esta forma se almacene esta información en disco duro.

1.8) Quita todas las ventanas desplegadas en el monitor.
1.9) Activa la frase "Genera nombres" en la pantalla.
1.10) Despliega la ventana -36
Nombres para diccionarios de datos.
Nombres para ambientes de programación.
1.11) Activa el renglon "Nombres para ambientes de programación" de esta ventana.

Estado 2:
Acciones:
2.1) Quita la última ventana desplegada en el monitor.
2.2) Activa la frase "Genera nombres" de la pantalla.
2.3) Construye ventana -36
Nombres primarios.
Nombres de programación.
2.4) Activa renglón "Nombres primarios"
de esta ventana.

Estado 3:
Acciones:
3.1) Quita la última ventana desplegada en el monitor.
3.2) Activa la palabra "Genera archivos" de la pantalla.
3.3) Construye ventana -46
Genera reportes
3.4) Activa renglón "Genera reportes"
de esta ventana.

Estado 4:
Acciones:
4.1) Quita la última ventana desplegada en el monitor.
4.2) Activa la palabra "Salir" de la pantalla.
4.3) Construye ventana -37
A la descripción de datos
4.4) Activa renglón "A la descripción de datos"
de esta ventana.

Estado 5:
Acciones:
5.1) Quita la última ventana desplegada en el monitor.
5.2) Activa la palabra "Info" de la pantalla.
5.3) Construye ventana 11
Autor
Memoria disponible
5.4) Activa renglón "Autor"
de esta ventana.

Estado 6:
Acciones:
6.1) Quita todas las ventanas desplegadas en el monitor.
6.2) Estado terminal del autómata.

Estado 7:
  Acciones:
  7.1) Activa renglón "Nombres primarios"
       de la ventana desplegada en el estado 2

Estado 8:
  Acciones:
  8.1) Activa renglón "Nombres de programación"
       de la ventana desplegada en el estado 2.

Estado 9:
  Acciones:
  9.1) Activa bandera brinca:=verdadero.

Estado 10:
  Acciones:
  10.1) Activa renglón "Autor"
        de la ventana desplegada en el estado 5.

Estado 11:
  Acciones:
  11.1) Activa renglón "Memoria disponible"
        de la ventana desplegada en el estado 5.

Estado 12:
  Acciones:
  12.1) Despliega ventana con la información del autor.
  12.2) Llama a la rutina que se encarga de desplegar la
        fecha y hora del sistema.
  12.3) Genera una espera hasta que se oprima cualquier
        tecla, en cuyo caso se quita a la ventana de la
        acción 12.1
  12.4) Asigna al estado actual del autómata el valor de
        "estado 5".

Estado 13:
  Acciones:
  13.1) Construye ventana para presentación de la memoria
        disponible del sistema.
  13.2) Despliega la información sobre la memoria disponible
        (en bytes) en el sistema.
  13.3) Genera una espera hasta que se oprima cualquier
        tecla, en cuyo caso se quita a la ventana de la
        acción 13.1
  13.4) Asigna al estado actual del autómata el valor de
        "estado 5".

Estado 14:
  Acciones:
  14.1) Llama al autómata que se encarga de controlar las
        ventanas y teclado usados en la definición de los
        documentos de reportes del sistema.
  14.2) Construye ventana -46

```
        Genera reportes
  14.3) Activa renglón "Genera reportes"
        de esta ventana.
  14.4) Activa bandera brinca:=verdadero.
```

Función de transición de estados:

```
 if not(fin_ed_abrev) then
   begin
   if not(brinca) then
     begin
     ed_abrev_pasada:=ed_abrev_actual;
     ed_abrev_sigte:=mat_abrev[ed_abrev_actual,ve_tecla];
     ed_abrev_actual:=ed_abrev_sigte;
     end
   else
     begin
     ed_abrev_actual:=ed_abrev_pasada;
     ed_abrev_sigte:=mat_abrev[ed_abrev_actual,ve_tecla];
     ed_abrev_actual:=ed_abrev_sigte;
     brinca:=false;
     end;
   end;
 until (ed_abrev_actual<0)or(fin_ed_abrev);
```

Matriz de transición de estados del autómata:

```
     for iy:=0 to 20 do
       for ix:=0 to 5 do
         mat_abrev[iy,ix]:=9;

         mat_abrev[0,0]:=3; mat_abrev[0,1]:=5;
         mat_abrev[0,2]:=0; mat_abrev[0,4]:=8;
         mat_abrev[0,5]:=7;

         mat_abrev[1,0]:=3; mat_abrev[1,1]:=5;
         mat_abrev[1,2]:=1; mat_abrev[1,4]:=8;
         mat_abrev[1,5]:=7;

         mat_abrev[2,0]:=3; mat_abrev[2,1]:=5;
         mat_abrev[2,2]:=0; mat_abrev[2,4]:=8;
         mat_abrev[2,5]:=7;

         mat_abrev[3,0]:=4; mat_abrev[3,1]:=2;
         mat_abrev[3,2]:=14;

         mat_abrev[4,0]:=5; mat_abrev[4,1]:=3;
         mat_abrev[4,2]:=6;

         mat_abrev[5,0]:=2; mat_abrev[5,1]:=4;
```

```
mat_abrev[5,2]:=12;mat_abrev[5,4]:=11;
mat_abrev[5,5]:=11;

mat_abrev[7,0]:=4; mat_abrev[7,1]:=5;
mat_abrev[7,2]:=0; mat_abrev[7,4]:=8;
mat_abrev[7,5]:=8;

mat_abrev[8,0]:=4; mat_abrev[8,1]:=5;
mat_abrev[8,2]:=1; mat_abrev[8,4]:=7;
mat_abrev[8,5]:=7;

mat_abrev[10,0]:=2; mat_abrev[10,1]:=4;
mat_abrev[10,2]:=12;mat_abrev[10,4]:=11;
mat_abrev[10,5]:=11;

mat_abrev[11,0]:=2; mat_abrev[11,1]:=4;
mat_abrev[11,2]:=13;mat_abrev[11,4]:=10;
mat_abrev[11,5]:=10;

mat_abrev[14,0]:=4; mat_abrev[14,1]:=2;
mat_abrev[14,2]:=14;
```

5.4.8.- Descripción del autómata usado para la definición de la estructura de los documentos de reportes generados por el sistema.

La descripción de los autómatas del sistema, está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción de los estados: Esta sección contiene la explicación de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habrá de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección se muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción de los estados del autómata :

Estado inicial : estado 2
Despliega ventana (-49). Esta ventana se usa para que el sistema muestre, conforme se seleccionen, los diferentes reportes tipo que formarán parte del documento a generar.

A continuación se describen las acciones efectuadas por cada

uno de los estados de este autómata:

Estado 0:
  Acciones:
  0.1) Estado terminal del autómata.

Estado 1:
  Acciones:
  1.1) Habilita bandera brinca:=verdadero.

Estado 2:
  Acciones:
  2.1) Despliega ventana -47
       'Selección del documento a generar   '
       'Reporte de palabras clase   '
       'Reporte de palabras entidad   '
       'Reporte de diccionarios de datos   '
       'Reporte de ambientes de programación   '
       'Reporte de reglas de abreviación   '
       'Reporte de abreviaciones especiales   '
       'Reporte de eliminaciones   '
       'Reporte de nombres generados en sesión'
  2.2) Activa renglón "Selección del documento a generar"
       de esta ventana.
  2.3) Despliega ventana -48
       'Nombre del documento: '
  2.4) Si se definió con anterioridad a otro documento, llama
       a la rutina que se encarga de escribir al buffer de
       documentos en disco duro. De esta forma se garantiza
       que el documento anterior se genere antes del nuevo.
  2.5) Llama a la rutina que hace el análisis del teclado
       para leer el nombre del nuevo documento a generar.
  2.6) Si se oprimió la tecla ESC entonces no se genera
       reporte.
  2.7) Quita la última ventana desplegada en el monitor.
  2.8) Limpia la lista de reportes tipo.
  2.9) Almacena en la lista de reportes tipo al reporte de
       nombres generados en sesión.

Estado 3:
  Acciones:
  3.1) Activa el renglón "Selección del documento a generar"
       de la ventana desplegada en el estado 2.

Estado 4:
  Acciones:
  4.1) Activa el renglón "Reporte de palabras clase"
       de la ventana desplegada en el estado 2.

Estado 5:
  Acciones:
  5.1) Activa el renglón "Reporte de palabras entidad"
       de la ventana desplegada en el estado 2.

Estado 6:
Acciones:
6.1) Activa el renglón "Reporte de diccionarios de datos" de la ventana desplegada en el estado 2.

Estado 7:
Acciones:
7.1) Activa el renglón "Reporte de ambientes de programación" de la ventana desplegada en el estado 2.

Estado 8:
Acciones:
8.1) Activa el renglón "Reporte de reglas de abreviación" de la ventana desplegada en el estado 2.

Estado 9:
Acciones:
9.1) Activa el renglón "Reporte de abreviaciones especiales" de la ventana desplegada en el estado 2.

Estado 10:
Acciones:
10.1) Activa el renglón "Reporte de eliminaciones" de la ventana desplegada en el estado 2.

Estado 11:
Acciones:
11.1) Activa el renglón "Reporte de nombres generados en sesión" de la ventana desplegada en el estado 2.

Estado 12:
Acciones:
12.1) Despliega ventana -48
'Nombre del documento: '
12.2) Si se definió con anterioridad a otro documento, llama a la rutina que se encarga de escribir los documentos en disco duro. De esta forma se garantiza que el documento anterior se genere antes del nuevo.
12.3) Llama a la rutina que hace el análisis del teclado para leer el nombre del nuevo documento a generar.
12.4) Si se oprimió la tecla ESC entonces no se genera reporte.
12.5) Quita la última ventana desplegada en el monitor.
12.6) Almacena en la lista de reportes tipo al reporte de nombres generados en sesión.

Estado 13:
Acciones:
13.1) Almacena en la lista de reportes tipo, al reporte de palabras clase. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
13.2) Activa bandera brinca

Estado 14:
Acciones:
14.1) Almacena en la lista de reportes tipo, al reporte de palabras entidad. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
14.2) Activa bandera brinca.

Estado 15:
Acciones:
15.1) Almacena en la lista de reportes tipo, al reporte de diccionarios de datos. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
15.2) Activa bandera brinca.

Estado 16:
Acciones:
16.1) Almacena en la lista de reportes tipo, al reporte de ambientes de programación. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
16.2) Activa bandera brinca.

Estado 17:
Acciones:
17.1) Almacena en la lista de reportes tipo, al reporte de reglas de abreviación. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
17.2) Si se dió de alta al reporte de reglas de abreviación en la lista de reportes tipo, entonces se llama al autómata auxiliar encargado de controlar ventanas para la selección de diccionarios de datos y ambientes de programación para los cuales se desea presentar sus respectivas reglas de abreviación.
17.3) Activa bandera brinca.

Estado 18:
Acciones:
18.1) Almacena en la lista de reportes tipo, al reporte de abreviaciones especiales . Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
18.2) Activa bandera brinca.

Estado 19:
Acciones:
19.1) Almacena en la lista de reportes tipo, al reporte de eliminaciones. Si ya existe dicho reporte en la lista, entonces se elimina de ésta.
19.2) Activa bandera brinca.

Estado 20:
Acciones:
20.1) Elimina de la lista de reportes tipo, al reporte de nombres generados en sesión. Si no existe dicho reporte en la lista, entonces se almacene en ésta.
20.2) Activa bandera brinca.

Función de transición de estados:

```
    if not(fin_autom_rep) then
      begin
        if not(brinca) then
          begin
            ed_rep_pasada:=ed_rep_actual;
            ed_rep_sigte:=mat_rep[ed_rep_actual,sig_tecla];
            ed_rep_actual:=ed_rep_sigte;
          end
        else
          begin
            ed_rep_actual:=ed_rep_pasada;
            ed_rep_sigte:=mat_rep[ed_rep_actual,sig_tecla];
            ed_rep_actual:=ed_rep_sigte;
            brinca:=false;
          end;
      end;
  until (ed_rep_actual<0)or(fin_autom_rep);
  quita_todas;
  if not(bndra_imprime_nombres) then gen_reporte;
  end;
```

Matriz de transición de estados para el autómata:

```
mat_rep[ 2,2]:=12; mat_rep[ 2,3]:= 0;
mat_rep[ 2,4]:= 4; mat_rep[ 2,5]:=11;
mat_rep[ 3,2]:=12; mat_rep[ 3,3]:= 0;
mat_rep[ 3,4]:= 4; mat_rep[ 3,5]:=11;
mat_rep[ 4,2]:=13; mat_rep[ 4,3]:= 0;
mat_rep[ 4,4]:= 5; mat_rep[ 4,5]:= 3;
mat_rep[ 5,2]:=14; mat_rep[ 5,3]:= 0;
mat_rep[ 5,4]:= 6; mat_rep[ 5,5]:= 4;
mat_rep[ 6,2]:=15; mat_rep[ 6,3]:= 0;
mat_rep[ 6,4]:= 7; mat_rep[ 6,5]:= 5;
mat_rep[ 7,2]:=16; mat_rep[ 7,3]:= 0;
mat_rep[ 7,4]:= 8; mat_rep[ 7,5]:= 6;
mat_rep[ 8,2]:=17; mat_rep[ 8,3]:= 0;
mat_rep[ 8,4]:= 9; mat_rep[ 8,5]:= 7;
mat_rep[ 9,2]:=18; mat_rep[ 9,3]:= 0;
mat_rep[ 9,4]:=10; mat_rep[ 9,5]:= 8;
mat_rep[10,2]:=19; mat_rep[10,3]:= 0;
mat_rep[10,4]:=11; mat_rep[10,5]:= 9;
mat_rep[11,2]:=20; mat_rep[11,3]:= 0;
mat_rep[11,4]:= 3; mat_rep[11,5]:=10;
mat_rep[12,2]:=12; mat_rep[12,3]:= 0;
mat_rep[12,4]:= 4; mat_rep[12,5]:=11;
```

Descripción del autómata auxiliar:

Estados del autómata auxiliar:

Estado inicial: 2

Despliega ventana -51. Esta ventana es usada por el sistema para mostrar, conforme se seleccionen, los nombres de diccionarios de datos y ambientes de programación cuyas reglas de abreviación aparecerán en el documento a generar.

Estado inicial: estado 2.

A continuación se muestran las acciones realizadas por cada uno de los estados de este autómata:

Estado 0:
  Acciones:
  0.1) Estado terminal del autòmata auxiliar.

Estado 1:
  Acciones:
  1.1) Activa bandera brinca_aux:=verdadero.

Estado 2:
  Acciones:
  2.1) Despliega ventana -50:
       Reglas de diccionarios de datos
       Reglas de ambientes de programación
  2.2) Activa renglón "Reglas de diccionarios de datos"
       de esta ventana.

Estado 3:
  Acciones:
  3.1) Activa renglón "Reglas de diccionarios de datos"
       de la ventana desplegada en el estado 2

Estado 4:
  Acciones:
  4.1) Activa renglón "Reglas de ambientes de programación"
       de la ventana desplegada en el estado 2

Estado 5:
  Acciones:
  5.1) Activa bandera brinca_aux:=verdadero.
  5.2) Llama a la rutina encargada de manejar la consulta a las listas del sistema. Lista consultada: Lista de diccionarios de datos.
  5.3) Llama a la rutina encargada de analizar al teclado para aceptar el nombre del diccionario de datos cuyas reglas quieren incluirse en la lista de reglas de

abreviación que aparecerán en el reporte.

5.4) Almacena el nombre del diccionario de datos en la lista de reglas de abreviacion que aparecerán en el reporte. Si el nombre ya existia con anterioridad, entonces se elimina de esta lista.

Estado 6:

Acciones:

6.1) Activa bandera brinca_aux:=verdadero.

6.2) Llama a la rutina encargada de manejar la consulta a las listas del sistema. Lista consultada: Lista de ambientes de programación.

6.3) Llama a la rutina encargada de analizar al teclado para aceptar el nombre del ambiente de programación cuyas reglas quieren incluirse en la lista de reglas de abreviación que aparecerán en el reporte.

6.4) Almacena el nombre del diccionario de datos en la lista de reglas de abreviación que aparecerán en el reporte. Si el nombre ya existia con anterioridad, entonces se elimina de esta lista.

Función de transición de estados del autómata auxiliar:

```
if not(fin_autom_aux_rep) then
  begin
    if not(brinca_aux) then
      begin
        ed_aux_rep_pasada:=ed_aux_rep_actual;
        ed_aux_rep_sigte:=
        mat_aux_rep[ed_aux_rep_actual,sig_tecla];
        ed_aux_rep_actual:=ed_aux_rep_sigte;
      end
    else
      begin
        ed_aux_rep_actual:=ed_aux_rep_pasada;
        ed_aux_rep_sigte:=
        mat_aux_rep[ed_aux_rep_actual,sig_tecla];
        ed_aux_rep_actual:=ed_aux_rep_sigte;
        brinca_aux:=false;
      end;
  end;
until (ed_aux_rep_actual<0)or(fin_autom_aux_rep);
quita_ventana;
quita_ventana;
end;
```

Matriz de transición de estados del autómata auxiliar para la definición de la estructura de documentos generados por el sistema.

```
mat_aux_rep[ 2,2]:=5 ;mat_aux_rep[ 2,3]:= 0;
mat_aux_rep[ 2,4]:=4 ;mat_aux_rep[ 2,5]:= 4;
mat_aux_rep[ 3,2]:=5 ;mat_aux_rep[ 3,3]:= 0;
mat_aux_rep[ 3,4]:=4 ;mat_aux_rep[ 3,5]:= 4;
mat_aux_rep[ 4,2]:=6 ;mat_aux_rep[ 4,3]:= 0;
mat_aux_rep[ 4,4]:=3 ;mat_aux_rep[ 4,5]:= 3;
```

5.4.9.- Descripción del autómata encargado de realizar el análisis morfológico de las palabras que forman parte de la definición del dato. (Análisis de sustantivos y adjetivos)

La descripción de este autómata está constituida por las siguientes secciones, las cuales se explican a continuación:

Descripción general : Indica cuál es el propósito del autómata, y sus principales características.

Descripción de los estados: Esta sección contiene la explicación de todos los estados que forman parte del autómata. Esta sección está formada por : **Estado** y sus correspondientes **acciones.**

Estado : Esta subsección indica el número del estado que habrá de explicarse.

Acciones : Esta subsección explica todas las acciones que se efectúan en el estado que se describe.

Función de transición de estados : Esta sección explica el algoritmo utilizado para calcular la función de transición de estados usada en el presente autómata.

Matriz de transición de estados : Esta sección muestra el contenido de los elementos de la matriz de transición de estados usada en el presente autómata.

A continuación se describe al autómata siguiendo el formato anterior:

Descripción general:

El autómata recibe 2 parámetros:
- La palabra a ser analizada
- La categoría sintáctica a la que se espera que corresponda la palabra a ser analizada. También se le conoce a este parámetro como "tipo

esperado".

Como respuestas posibles, el autómata genera las siguientes:

"no corresponde al tipo"	para indicar que la palabra analizada no corresponde a la categoría sintáctica especificada.
"corresponde al tipo"	para indicar que la palabra analizada sí corresponde a la categoría sintáctica especificada.
"no se puede afirmar nada"	para indicar que el análisis no puede determinar la naturaleza de la palabra.

Antes de iniciar la ejecución del autómata, se invierte el orden de las letras que forman a la palabra que se va a analizar.

Descripción de los estados del autómata

Estado inicial: Estado #3

Estado 0:
Acciones:
0.1) Activa bandera de fin de ejecución fin:=verdadero.
0.2) Respuesta del análisis: "No corresponde al tipo".

Estado 1:
Acciones:
1.1) Activa bandera de fin de ejecución fin:=verdadero.
1.2) Si la longitud de la palabra es mayor al número de caracteres analizados, entonces queda restante una porción de la palabra, por lo que es válido el análisis. Si éste es el caso, entonces:

1.2.1 Si el tipo de palabra esperado es "sustantivo" entonces la respuesta del análisis es: "corresponde al tipo".
1.2.2 Si el tipo de palabra esperado es "adjetivo" entonces la respuesta del análisis es: "no se puede afirmar nada". Esto quiere decir que se ha identificado el gramema característico de un sustantivo cuando en realidad se esperaba que la palabra fuera un adjetivo.

Estado 2:
Acciones:
2.1) Activa bandera de fin de ejecucion fin:=verdadero.
2.2) Si la longitud de la palabra analizada es mayor al

número de caracteres analizados, entonces queda una porción restante de la palabra, por lo que el análisis es válido. Si tal es el caso, entonces:

2.2.1 Si el tipo de palabra esperado es "adjetivo" entonces la respuesta del análisis es: "corresponde al tipo".
2.2.2 Si el tipo de palabra esperado es "sustantivo" entonces la respuesta del análisis es: "no se puede afirmar nada". Esto quiere decir que se ha identificado el gramema característico de un sustantivo cuando en realidad se esperaba que la palabra correspondiera a un adjetivo.

Los estados #3 al #44 no realizan ninguna acción. Sirven para hacer secuencias de transiciones (en las que se llamará a la rutina "sig_letra" que traerá la siguiente letra de la palabra) hasta alcanzar alguna transición que lleve al autómata a los estados finales #0, ó #1, ó #2.

Función de transicion de estados:

Si no se detectó ningún error, entonces:

edo_del_autómata:= M [edo_del_autómata,sig_letra]

en donde :

M	Representa a la matriz de transición de estados.
edo_del_autómata	Representa al estado actual.
sig_letra	Es una función que realiza las siguientes operaciones:

Si el número de caracteres analizados llega a ser igual o mayor que la longitud de la palabra analizada, entonces existe un error. En tal caso, se llevan a cabo las siguientes acciones:

- El autómata cambia al estado #0.
- Se determina el fin de la ejecución del autómata.
- Se activa la bandera de error detectado.

Si no se ha alcanzado la longitud total de la palabra, quiere decir que la palabra no es sólo un gramema, y por lo tanto la función entrega como resultado la siguiente letra de la palabra analizada. Recuérdese que la palabra fue previamente invertida. Esto se traduce a estar analizando a la palabra de su extremo derecho hacia la izquierda. Esto se debe a que los gramemas característicos de sustan-

tivos y adjetivos se localizan en el extremo derecho.

Matriz de transición de estados:

La matriz de transición de estados se inicializa al valor '0', para que de esta forma se llegue al estado final 0. Esto quiere decir que una secuencia de caracteres que no corresponda a alguno de los gramemas hará que el autómata llegue al estado 0. Recuérdese que el estado 0 es un estado que indica que no se reconoció a la palabra como sustantivo o como adjetivo.

Los renglones corresponden a los estados del autómata, mientras que las columnas corresponden a las letras del alfabeto español:

```
for i:=1 to 50 do
  for j:=ord('a') to ord('z') do
    M[i,(chr(j))]:=0;
```

A continuación se muestra el contenido de la matriz. La forma en que se ha escrito permite ver las secuencias de caracteres que deben presentarse para que el autómata reconozca la presencia de un gramema. En la columna derecha se muestra al gramema reconocido (nota: el gramema se escribió en orden normal, de izquierda a derecha).

```
M[ 3,a]:=4;  M[ 4,c]:=5;  M[ 5,i]:=1;                  (*ica *)
             M[ 4,d]:=6;  M[ 6,a]:=1;                  (*ada *)
                          M[ 6,e]:=1;                  (*eda *)
                          M[ 6,i]:=1;                  (*ida *)
             M[ 4,i]:=1;                               (*ia  *)
             M[ 4,r]:=1;                               (*ra  *)
             M[ 4,t]:=7 ; M[ 7,a]:=1;
                          M[ 7,s]:=8;  M[ 8,i]:=1;     (*ista*)
             M[ 4,z]:=9 ; M[ 9,e]:=1;                  (*eza *)
                          M[ 9,n]:=10; M[10,a]:=1;     (*anza*)
             M[ 4,n]:=11; M[11,i]:=1;                  (*ina *)
             M[ 4,s]:=12; M[12,e]:=1;                  (*esa *)
                          M[12,i]:=1;                  (*isa *)

M[ 3,d]:=13; M[13,a]:=1;                               (*ad  *)
             M[13,u]:=14; M[14,t]:=1;                  (*tud *)

M[ 3,e]:=15; M[15,i]:=1;                               (*ie  *)
             M[15,j]:=16; M[16,a]:=1;                  (*aje *)
             M[15,l]:=34; M[34,b]:=2;                  (*ble *)
             M[15,r]:=35; M[35,b]:=2;                  (*bre *)
                          M[35,t]:=36; M[36,s]:=37;
                                       M[37,e]:=2; (*estre*)
```

```
                M[15,t]:=38; M[38,n]:=39; M[39,a]:=1;  (*ante*)
                                          M[39,e]:=2;  (*ente*)
                             M[38,s]:=40; M[40,e]:=2;  (*este*)

M[ 3,l]:=41; M[41,a]:=2;                               (*al  *)
                M[41,i]:=2;                            (*il  *)


M[ 3,n]:=17; M[17,o]:=18; M[18,i]:=1;                  (*ion *)
                M[17,e]:=19; M[19,m]:=1;               (*men *)

M[ 3,o]:=20; M[20,d]:=21; M[21,a]:=1;                  (*ado *)
                          M[21,e]:=1;                  (*edo *)
                          M[21,i]:=1;                  (*ido *)
                M[20,g]:=22; M[22,z]:=23; M[23,a]:=1;  (*azgo*)
                M[20,i]:=1;                            (*io  *)
                M[20,m]:=24; M[24,s]:=25; M[25,i]:=1;  (*ismo*)
                M[20,r]:=1;                            (*ro  *)
                M[20,t]:=27; M[27,a]:=1;               (*ato *)
                             M[27,n]:=28; M[28,e]:=29;
                                          M[29,i]:=30;
                                          M[30,m]:=1;(*miento*)
                                          M[29,m]:=1 (*mento *)
                M[20,z]:=31; M[31,a]:=1;               (*azo *)
                M[20,c]:=42; M[42,i]:=2;               (*ico *)
                M[20,s]:=43; M[43,o]:=2;               (*oso *)
                M[20,v]:=44; M[44,i]:=2;               (*ivo *)


M[ 3,r]:=32; M[32,a]:=1;                               (*ar  *)
                M[32,o]:=1;                            (*or  *)

M[ 3,s]:=3;                                            (*s   *)

M[ 3,z]:=33; M[33,a]:=2;                               (*az  *)
                M[33,e]:=1;                            (*ez  *)
                M[33,i]:=1;                            (*iz  *)
```

5.4.10.- Relaciones existentes entre los diferentes autómatas del sistema:

Al diseñar al sistema, se consideró que éste debería ser lo suficientemente flexible para ayudar al administrador de datos a estructurar correctamente las definiciones de los datos, al mismo tiempo debería detectar errores y ayudar a corregirlos a través de una serie de diálogos con el usuario.

Se consideró que el manejo de las ventanas (realizar ciertas acciones a partir del teclado, dependiendo de la ventana que estuviera desplegada) podría hacerse a través de autómatas. También se consideró que el uso de autómatas era fundamental para poder controlar la aplicación del lenguaje de nomenclatura de datos y el control de los diálogos de ayuda en caso de detectarse un error.

Sin embargo, fue fundamental considerar que utilizar un sólo autómata para llevar el control de las funciones anteriores, sería muy complicado y por tanto el éxito del desarrollo del sistema estaría sujeto a un alto riesgo.

Por lo tanto, se determinó que el sistema debería estar compuesto por diferentes autómatas relacionados entre sí. Las ventajas de esta solución, son:

1) Cada autómata pudo desarrollarse en forma independiente, por lo que el sistema pudo evolucionar en forma de modelos. Estos modelos o prototipos

evolucionaron conforme se añadían los nuevos autómatas. Cada prototipo podía funcionar correctamente y proveer resultados aún cuando no existieran la totalidad de los autómatas.

2) Cada autómata pudo probarse en forma independiente; por lo tanto, si se detectaba un error era posible determinar cuál autómata lo originaba.

El propósito de esta sección es mostrar la forma en que se encuentran relacionados estos autómatas, y algunas características que tienen para poderlos manejar correctamente. Cada autómata está descrito completamente en las secciones anteriores.

El autómata **"autom_sintaxis"** se usa para la aplicación del lenguaje nomenclatura de datos (véase sección 5.4.2). Una característica de este autómata es que no hace transiciones a partir de elementos terminales del lenguaje, sino que espera a que otras rutinas reciban los elementos terminales, e identifiquen su naturaleza sintáctica. Una vez determinado el elemento no terminal (categoría sintáctica) el autómata hace la transición de un estado a otro hasta alcanzar el estado final.

Lo descrito en el párrafo anterior, se hace de la siguiente forma:

En cada estado del sistema se llama a la rutina

"lee_dato", la cual se encarga de recibir los caracteres que corresponden a la teclas oprimidas por el usuario. La cadena de caracteres o palabra es analizada inmediatemente por una rutina conocida con el nombre de **"semántica"**, la cual determina si dicha palabra corresponde al tipo de palabra esperado en el estado actual del autómata. Esta rutina se construyó a partir de las reglas descritas en la sección 5.4.3. (Reglas para llevar a cabo el análisis sintáctico y semántico en la definición del dato a nombrar). Esta rutina genera un código que es tomado por la función de transición de estados, de forma tal que dependiendo de dicho código y del estado actual, el autómata cambiará a otro estado.

A su vez, la rutina **"semántica"** utiliza a otro autómata. Este autómata se encarga de hacer el análisis morfológico cuando se trata de investigar si la palabra analizada es un sustantivo o un adjetivo. Este autómata se encuentra descrito en la sección 5.4.9.

Ahora bien, si la rutina **"semántica"** ha detectado un error (la palabra especificada no corresponde a la categoria sintáctica esperada en el estado actual), entonces el autómata deberá generar una serie de diálogos para dar solución al error. Para ello, en el autómata existe el estado "error", el cual pone en ejecución al autómata **"autom_error"**. Una vez corregido el error, el autómata **"autom_sintaxis"** recupera el estado anterior a la detección del error y continúa su ejecución normal. Para poder recuperar el

control y continuar su ejecución, el autómata **"autom_error"** actualiza la matriz de transición de estados del autómata **"autom_sintaxis"** con el valor:

```
mat_sintaxis[-1,0]:=edo_anterior;
```

Es lógico pensar que deben existir diferentes diálogos para diversos tipos de errores. El autómata **"autom_error"**, (descrito en la sección 5.4.5.) se encarga del manejo de las ventanas de diálogo con el usuario. Este autómata es muy especial, pues **no tiene un estado inicial único.** Dependiendo del tipo de error que se detecte, será diferente el estado inicial del autómata. Esto es una característica que le da cierta inteligencia al sistema, pues de esta forma establece diálogos flexibles con el usuario.

Para determinar en el momento de ejecución cuál será el estado inicial del autómata, se utilizó una función cuyo resultado corresponde al estado que será el estado inicial del autómata.

También el autómata **"autom_sintaxis"** se encuentra relacionado con el autómata **"autom_menus"**, que se encarga del manejo de las ventanas del menú principal (dicho autómata está descrito en la sección 5.4.4.). La forma en que se establece la relación, es la siguiente:

En cada estado del autómata **"autom_sintaxis"** se llama a rutina **"lee_dato"**, que está encargada de leer los caracteres

que corresponden a las teclas oprimidas por el usuario. Si esta rutina detecta que se ha oprimido a la tecla **ESC**, entonces contesta que debe ejecutarse al autómata **"autom_menus"**. Una vez que se ha terminado de ejecutar a este último autómata, entonces el control regresa a la rutina y posteriomente regresa al control al autómata **"autom_sintaxis"**.

A su vez, el autómata **"autom_menus"** activa a otros dos autómatas. Uno se le conoce como **"autom_reglas"** (descrito en la sección 5.4.6) y se encarga de manejar las ventanas para la definición y actualización de las reglas de abreviación del sistema. El otro se le conoce como **"autom_reps"** (descrito en la sección 5.4.8) y se encarga de la definición de la estructura de los documentos de reportes generados por el sistema.

Ahora bien, una característica importante del autómata **"autom_menus"** consiste en que su **matriz de transición de estados puede ser actualizada por el mismo autómata durante la ejecución del mismo.** Se determinó que esto pudiera hacerse para que el sistema pudiera tener otro rasgo de inteligencia. En este autómata, existe la activación de una rutina que se encaga de borrar palabras de la definición del dato. Es lógico que esta rutina no debería activarse si la definición del dato está vacía. Para ello, existe el estado #2 que maneja esta situación:

Estado 2:

Acciones:
2.1) Borra todas las ventanas desplegadas en el monitor.
2.2) Activa la frase 'Corregir texto' de la pantalla.
2.3) Si el estado actual del autómata principal del sistema es el estado 0, significa que no se ha escrito palabra alguna en la descripción del dato a nombrar. Por la razón anterior, no es posible corregir texto, lo cual se notifica al usuario. Se modifica la matriz de transición de estados del autómata de menús: mat_menú[2,2]:=6. De esta forma, si el usuario selecciona modificar texto y no existe texto qué modificar, se pasa al estado terminal del presente autómata.
2.4) Si el estado actual del autómata principal del sistema es diferente al estado 0, significa que si se ha escrito parte de la descripción del dato a nombrar. Se modifica la matriz de transición de estados del autómata de menús: mat_menú[2,2]:=21. De esta forma, si el usuario selecciona modificar texto, se pasa al estado en el cual se llama a la rutina encargada de modificar al texto de descripción del dato a nombrar.

Con esta actualización de la matriz de estados, el autómata sólo cambiará a uno de los 2 siguientes estados:

Estado 6:
Acciones:
6.1) Estado terminal del autómata.
6.2) Quita todas las ventanas desplegadas en el monitor.

Estado 21:
Acciones:
21.1) Quita todas las ventanas desplegadas en el monitor.
22.2) Llama a la rutina encargada de corregir el texto de descripción del dato a nombrar.
22.3) Estado final del autómata.

Cuando este autómata alcanza un estado final, termina su ejecución y regresa el control al autómata **"autom_sintaxis"**. Ahora bien, cuando el autómata **"autom_sintaxis"** llega al estado final (lo que significa que el usuario ha terminado de especificar la definición del dato), entonces el sistema cambiará de autómata y activará al autómata encargado de

controlar las ventanas que se usan en la fase de generación de nombres primarios y nombres de programación. Este autómata está descrito en la sección 5.4.7

5.5.- Diseño físico:

5.5.1.- Manejo del ambiente orientado a ventanas:

Al momento de diseñar a este sistema, se deseó que existiera un ambiente orientado a ventanas, en el que cada mensaje (ya fuese de ayuda, validación o corrección) se presentase en forma de ventanas que se sobreponen unas a otras dependiendo de la secuencia de mensajes que se quieren desplegar en el monitor. Asimismo, se deseó que las operaciones de consulta y actualización del acervo del sistema se hicieran también a través de ventanas, las cuales ofrecieran opciones que pudieran ser seleccionadas por el usuario. De esta forma, se consideró que el manejo del sistema sería sumamente amistoso y fácil de entender para los usuarios.

Ahora bien, contar con un ambiente orientado a ventanas, bajo las características mencionadas en los párrafos anteriores, requiere de ciertas estructuras de datos que den soporte a tal ambiente. A continuación se describen las estructuras de datos y mecanismos asociados a éstas para el manejo de las ventanas del sistema:

Para decidir el tipo de estructuras a usar en el sistema, se partió de las siguientes premisas:

1.- Debido a que la cantidad de ventanas a usar en el sistema podría ser muy grande (la versión actual cuenta con 85 ventanas diferentes), se consideró que deberían manejarse en memoria dinámica. Al

momento en que se requiriese que una ventana apareciera en el monitor, en ese momento el sistema se encargaría de generar a la ventana (mediante las rutinas que se describen en la sección 5.6.2). Dicha ventana quedaría presente en la memoria principal del sistema y la próxima vez que se requiriera desplegarla, el sistema sólo haría uso de las rutinas de graficación de ventanas.

2.- Al tener presentes ventanas cuyo propósito fuese proporcionar opciones que pudiera el usuario escoger para efectuar consultas y/o actualizaciones, se consideró que asociados al ambiente orientado a ventanas deberían existir ciertos mecanismos que se encargaran de desplegar o quitar ventanas, así como activar renglones de las ventanas dependiendo de ciertas secuencias de teclas que oprimiese el usuario. La forma en que se relacionarían estos mecanismos de despliegue y activación de ventanas con la secuencia de teclas, se hizo mediante el control proporcionado por autómatas descritos en las secciones 5.4.3 a 5.4.8.

3.- También se deseó que el sistema fuese capaz de llevar el control de la secuencia en la que desplegasen las ventanas del sistema, por lo que mediante este control deberían removerse las

ventanas en una secuencia inversa a la que se presentan en el monitor. Su comportamiento debería ser análogo al de una pila, en la que el último elemento añadido es el primero en removerse, y la secuencia de remoción de elementos es exactamente inversa a la secuencia de inserción.

Ahora bien, para satisfacer las premisas anteriores, se definieron las siguientes estructuras de datos:

- **vector de apuntadores a las ventanas.**
- **vector de parámetros físicos de las ventanas.**
- **pila de control de ventanas.**
- **apuntador a la última área desplegada.**

El vector de parámetros físicos de las ventanas, así como el **vector de apuntadores a las ventanas** son estructuras de datos cuya función consiste en proporcionar la información para la construcción de las ventanas en memoria dinámica así como para tener acceso a ellas al momento de desplegarlas en el monitor.

El **vector de parámetros físicos de las ventanas** contiene la información que requieren las rutinas de graficación de ventanas. Esta información es:

- coordenadas de la esquina superior izquierda de la ventana.
- longitud horizontal de la ventana.
- longitud vertical de la ventana.

- color de la ventana.

El **vector de apuntadores a las ventanas** es un vector de apuntadores, que al inicio de la ejecución del sistema, cada apuntador apunta a nil (es decir, no existen ventanas en memoria dinámica). Al momento de requerir una ventana, si ésta no existe en memoria dinámica, entonces una serie de rutinas se encargan de inicializar a uno de los apuntadores del vector y otras rutinas generan a la ventana deseada tomando como base a la información del vector de parámetros físicos de las ventanas. En la sección 5.6.2. se encuentra la descripción de todas las rutinas del sistema.

Una vez que se ha generado a la ventana, ésta queda apuntada por un elemento del vector de apuntadores; queda permanente en la memoria y las próximas veces que se requiera desplegarla, se tendrá acceso a ella mediante el elemento del vector que le hace referencia. El diagrama 5.5.1.1 muestra al vector de apuntadores a las ventanas y algunos de sus elementos que hacen referencia a algunas ventanas del sistema.

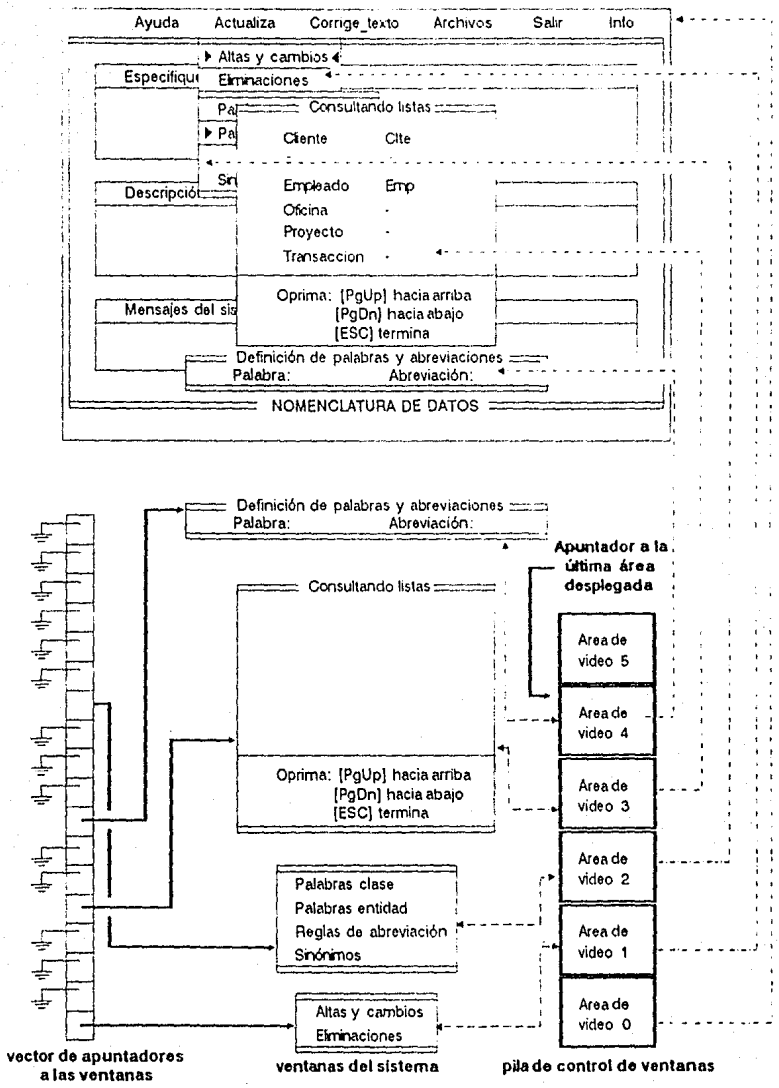


Diagrama 5.5.1.1

La **pila de control de ventanas** así como **el apuntador a la última área desplegada** son estructuras de datos cuya función consiste en proporcionar el control adecuado para desplegar y remover a las ventanas en el monitor.

Cada **elemento** de la **pila de control de ventanas** es un área de memoria que tiene las mismas características que el área de memoria destinada por el sistema operativo para la salida a video; es decir, cada elemento de la pila es una matriz de 25 x 80 elementos c/u consiste de 1 byte para el caracter ASCII y 1 byte para el color. El elemento en la base de la pila, es realmente la memoria destinada a video por el sistema operativo. El **apuntador a la última área desplegada** tiene la función de indicar cuál de estas matrices es la que contiene la información requerida para remover a la última ventada desplegada en el monitor, así como para indicar si no se están desplegando ventanas.

El elemento que se encuentra en la base de la pila es en realidad la memoria de video destinada por el sistema operativo para manejar las salidas hacia el monitor. Cualquier cambio que se quiera hacer al desplegar una ventana, deberá entonces actualizar a dicho elemento de la pila de control. Los demás elementos de la pila se usan en realidad para almacenar secciones de video que serán encimadas al desplegar ventanas. De esta forma, se cuenta con la información necesaria para recuperar las porciones de

video encimadas por las ventanas y así hacer el efecto de remover ventanas del monitor.

Al desplegar una ventana en el monitor, se llevan a cabo las siguientes operaciones:

1.- El **apuntador a la última área desplegada** se actualiza y se apunta a la siguiente área disponible en la **pila de control de ventanas**, la cual queda en el tope de la pila.

2.- El **vector de parámetros físicos de las ventanas** proporciona la información acerca de dónde se desplegará la ventana, así como sus dimensiones. Esta información se toma como referencia y entonces en el área de video que se encuentra en el tope de la pila se copia una sección del área de video de la base de la pila, de forma tal que dicha sección corresponde a lo que quedaría encimado por la ventana que se pretende desplegar.

Lo anterior tiene la intención de poder recuperar la sección que fue encimada por la ventana y que debe aparecer nuevamente en el monitor al momento de remover a la misma ventana. El diagrama 5.5.1.2. muestra las secciones de video que se almacenarían en los elementos de la pila de control al estar añadiendo ventanas en una secuencia dada.

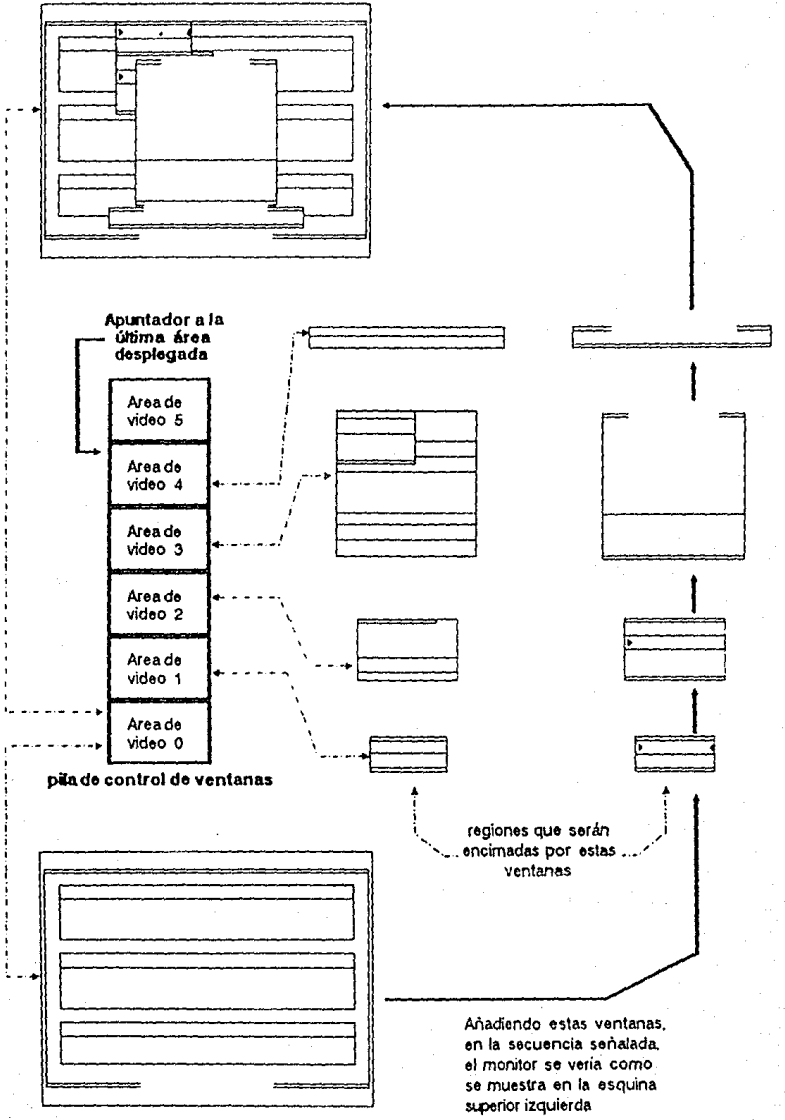
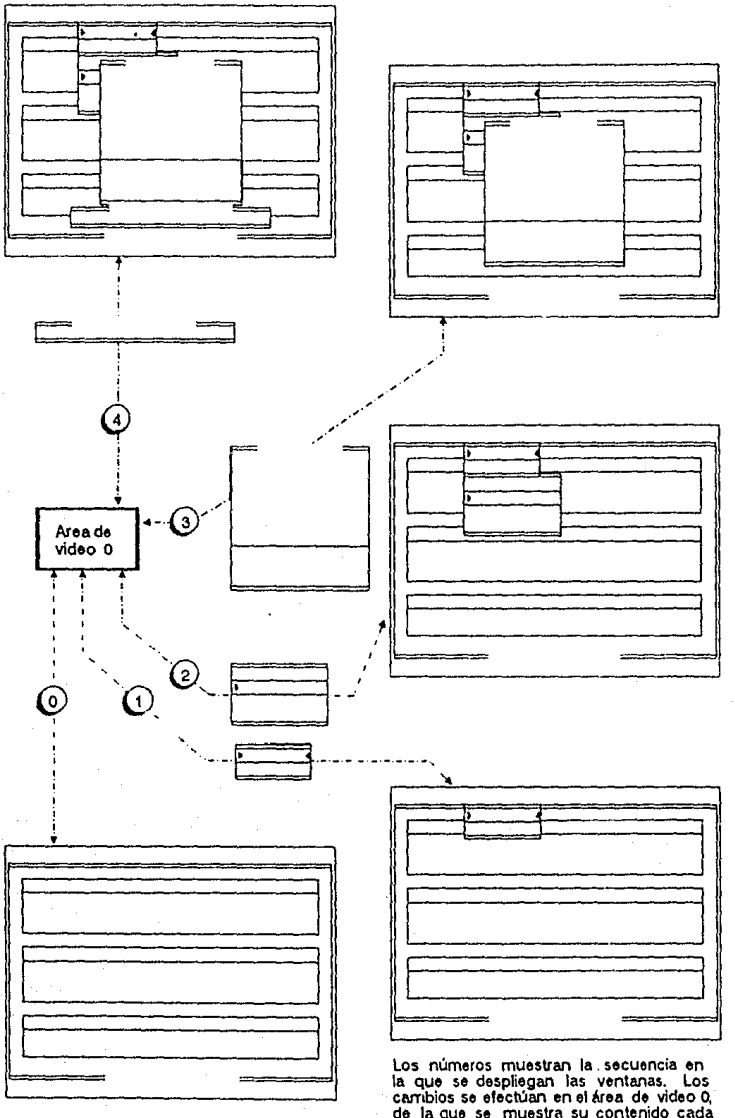


Diagrama 5.5.1.2

3.- Una vez que la sección que será encimada por la ventana a desplegar se ha copiado en el área de video que se encuentra en el tope de la pila de control, entonces las rutinas de graficación toman a la ventana a desplegar (la cual se encuentra en memoria dinámica) y su información correspondiente del vector de parámetros físicos y se encargan de copiarla en el área de video de la base de la pila de control. Al hacer esto, los cambios efectuados en el área de video de la base de la pila aparecen en el monitor.

El diagrama 5.5.1.3. muestra el contenido del área de video 0 conforme se despliegan ventanas, en una determinada secuencia.

4.- Al remover la última ventana desplegada, la información que se encuentra almacenada en el área de video del tope de la pila de control representa en realidad a la sección que se encuentra debajo de la ventana a remover. Con esta información, las rutinas que se encargan de remover a la ventana copian dicha sección en el área de video de la base de la pila y se despliega en el monitor. Con esta operación se da el efecto de remover la ventana, en realidad, lo que se hace es recuperar la sección que se encontraba debajo de la ventana desplegada.



Los números muestran la secuencia en la que se despliegan las ventanas. Los cambios se efectúan en el área de video 0, de la que se muestra su contenido cada vez que se añade una ventana.

Diagrama 5.5.1.3

Una vez hecho lo anterior, el área de video que se encuentra en el tope de la pila ya no sirve, por lo que se retira de la pila al actualizar al apuntador de la pila una posición hacia abajo. El área a la cual apunta se considerará ahora como la última área de video desplegada en el monitor.

5.5.2.- Manejo de estructuras de datos dinámicas:

De la sección 3.5.1. "Componentes de los estándares de nomenclatura de datos", así como de la red semántica del sistema de nomenclatura de datos (presentada en la sección 5.4.1.) se mostró que el acervo del sistema está formado por:

- Conjunto de palabras clase y abreviaturas
- Conjunto de palabras entidad y abreviaturas
- Conjunto de palabras modificadoras
- Conjunto de artículos y preposiciones
- Conjunto de nombres de diccionarios de datos
- Conjunto de nombres de ambientes de programación
- Conjunto de reglas de abreviación, por diccionario de datos y ambientes de programación.

Este acervo requirió de una serie de estructuras de datos que permitiera representarlo en la memoria principal de la microcomputadora. A continuación se muestran las premisas básicas que se consideraron para seleccionar las estructuras de datos usadas en el sistema:

1.- Si bien la cantidad de palabras clase y entidad es finita, es un hecho que en cada institución no se conoce de antemano, y que a lo largo de la implantación de la función de la administración de datos nuevas palabras pueden surgir. Debido a esta

razon, es claro que no podria usarse una estructura de datos estática para manejar a las palabras. Por otra parte, al añadir nuevas palabras es un hecho que su consulta requeriria un acceso rápido al acervo. Ademas, se considero primordial que el usuario pudiese consultar al acervo de palabras, por lo que se requeriria que las palabras del acervo quedasen ordenadas alfabéticamente.

2.- Existe una relación biunívoca entre las palabras del acervo y sus correspondientes abreviaturas, es decir, es incorrecto que dos palabras distintas tengan una misma abreviatura. Esta premisa requirió entonces considerar una serie de procesos de validación que deberian efectuarse al momento de dar de alta o actualizar a las palabras del acervo del sistema.

De acuerdo a las premisas anteriores, se decidió que las estructuras de datos más adecuadas serian **árboles binarios, existentes en memoria dinámica.** Las ventajas que proporcionan estas estructuras son las siguientes:

1.- La capacidad de los árboles binarios no está determinada por el espacio de memoria reservada por el compilador para estructuras de datos estáticas.

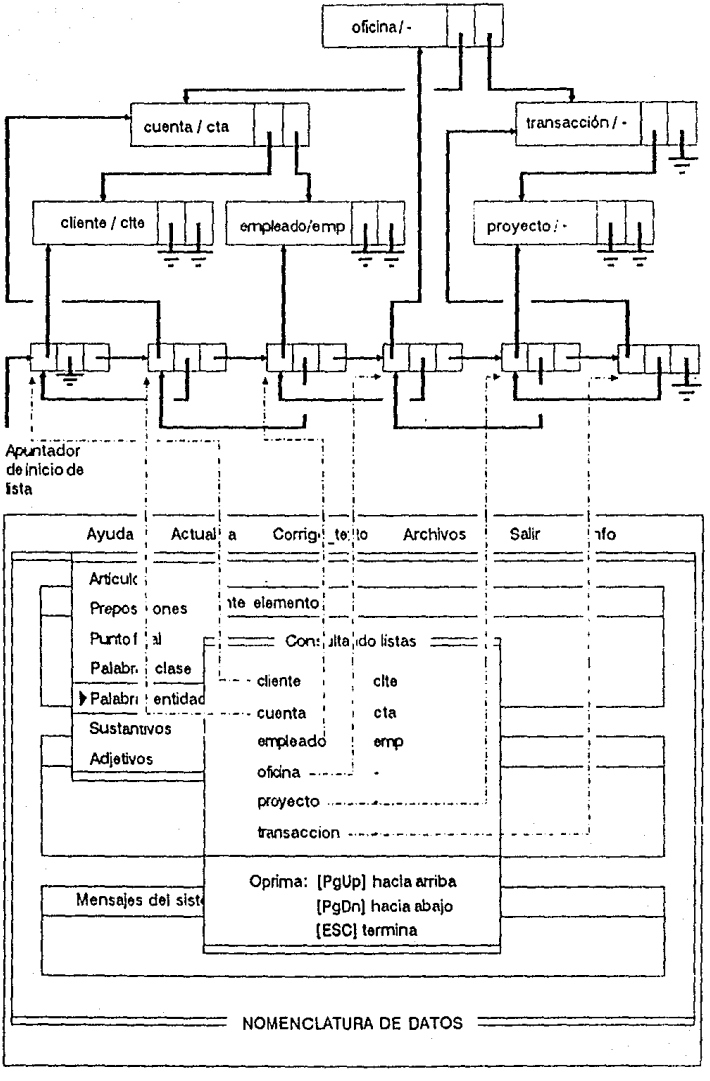
Precisamente al necesitar añadir un nuevo nodo al árbol binario, el espacio requerido se obtiene al momento de ejecución.

2.- Los nodos de un árbol binario tienen la característica de que al navegar al árbol en sentido **inorder**, da la apariencia de que los nodos se encuentran en orden. En realidad, al añadir nodos (sin importar la secuencia en que se añadan), siempre las rutinas de inserción harán que los nodos queden en la posición adecuada para que al momento de que se navegue al árbol, éstos aparezcan ordenados. De esta forma, no se necesitan utilizar rutinas que ordenen a los elementos.

3.- La explotación de la información de un árbol binario es mucho más eficiente que en otro tipo de estructuras, por ejemplo listas. Para tener acceso a la información de un nodo, en el peor de los casos deben examinarse $\log_2(n)$ nodos, cuando n es la cantidad total de nodos. Así, si se requiere obtener la información de uno de los nodos en un árbol de 1024 nodos, en el peor de los casos deberán examinarse 10 nodos para hallar la información, o bien, para indicar que no existe la información buscada.

Ahora bien, como se deseó que los usuarios pudieran consultar al acervo de palabras, es claro que dicha consulta se haría a través de una ventana del sistema. Como el espacio de la ventana sería limitado (es decir, sólo podrían desplegarse un número finito de elementos, faltando por desplegar al resto de elementos del total del árbol), fue necesario considerar que deberían existir rutinas que pudieran navegar **inorder** en bloques a los árboles. Esta navegación debería poderse hacer hacia adelante o hacia atrás. Debido a que las rutinas de navegación de árboles binarios que se usaron en el sistema son recursivas, se vió que era imposible satisfacer a la demanda anterior con sólo árboles binarios. Debido a lo anterior, asociados a los árboles binarios, se crearon **listas doblemente encadenadas.**

En estas listas doblemente encadenadas, cada nodo tiene 3 apuntadores: uno hace referencia al elemento siguiente de la lista (usado para navegación hacia adelante), otro hace referencia al elemento anterior de la lista (usado para navegación hacia atrás) y el otro hace referencia hacia su correspondiente nodo del árbol binario. Lo anterior se muestra en el diagrama 5.5.2.1. Mediante estas listas asociadas a los árboles binarios, es posible navegarlas hacia adelante o hacia atrás, y también navegarlas por bloques.



240 **Diagrama 5.5.2.1**

De acuerdo a lo anterior, existen las siguientes estructuras de datos:

- árbol binario de palabras clase y su correspondiente lista doblemente encadenada.

- árbol binario de palabras entidad y su correspondiente lista doblemente encadenada.

- árbol binario de nombres de diccionarios de datos y su correspondiente lista doblemente encadenada.

- árbol binario de nombres de ambientes de programación y su correspondiente lista doblemente encadenada.

Ahora bien, para satisfacer al requerimiento de que diferentes palabras no pueden tener la misma abreviación, existe otro árbol, en el que se almacenarán sólo a aquellas palabras que tengan asociadas abreviaturas convenidas. Esta situación se muestra en el diagrama 5.5.2.2.

Del análisis de morfemas característicos a adjetivos y sustantivos, se definió al autómata que se encargaria de reconocerlos (sección 5.3.7 "uso de gramemas y lexemas en el sistema de nomenclatura de datos"). Existen adjetivos y sustantivos que no serían reconocidos por dicho autómata.

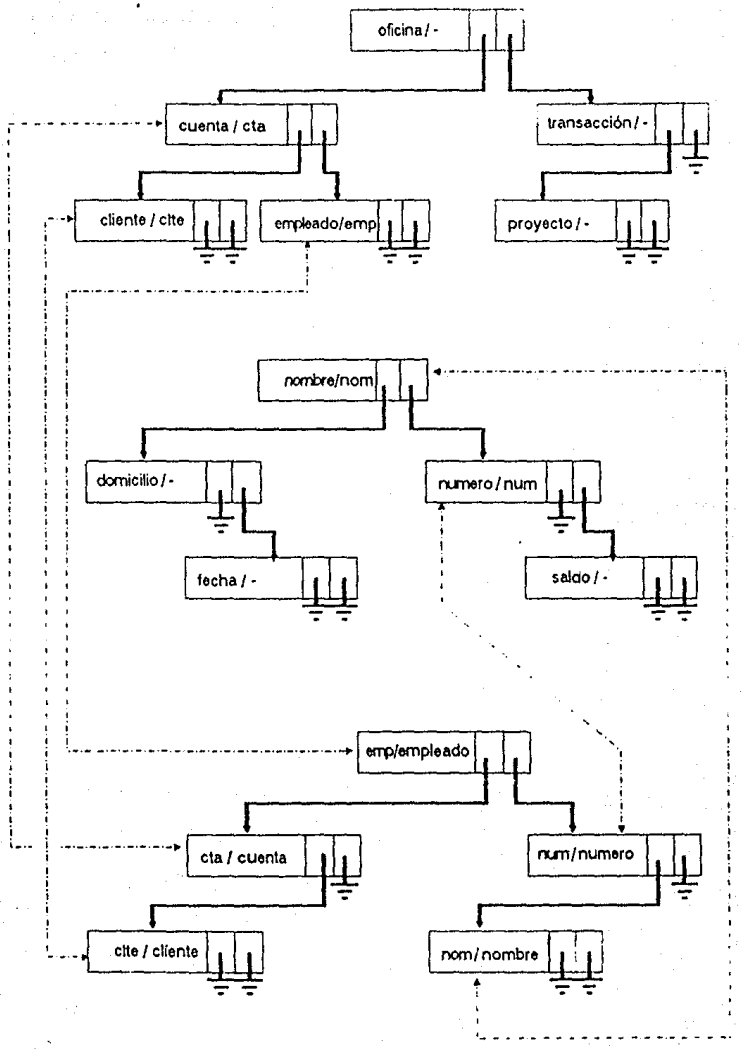


Diagrama 5.5.2.2

Para manejar estas excepciones, existen 2 arboles. Uno de ellos contendra sustantivos y el otro a los adjetivos. Al momento de ejecución del sistema, si una palabra no es reconocida como adjetivo o como sustantivo, el sistema usará a estos árboles para almacenarla como una excepción a los morfemas caracteristicos.

A continuación se muestran las politicas usadas en las operaciones de inserción, actualización y supresión de nodos de los árboles del sistema:

> Al momento de dar de alta un nodo en cualquiera de los árboles, se verifica que la llave del nodo ya exista en el árbol. Si la llave no existe en ningún nodo, entonces se trata de un nuevo nodo que se añade al árbol. Si la llave ya existe en algun nodo, entonces la información almacenada en dicho nodo es actualizada con la nueva información que se desea dar de alta.
>
> La supresión de nodos de un árbol, es una supresión lógica y no física. Para ello, en cada nodo de los árboles del sistema existe un campo que indica su estado. De esta forma, al momento de borrar un nodo, se colocará una marca de "eliminado" en dicho nodo, aunque físicamente exista en el árbol. Las rutinas de consulta, búsqueda e inserción toman en

cuenta dicha marca. Así, las rutinas de búsqueda y consulta indicarán que no existe un nodo eliminado, aunque físicamente sí exista. Las rutinas de reporte de elementos eliminados hacen la operación inversa: revisan a los nodos que contengan dicha marca y generan un reporte indicando cuáles nodos están marcados como eliminados.

Existen 2 razones por las cuales la eliminación de nodos de los árboles es lógica y no física. Estas razones son las siguientes:

Al eliminar nodos de un árbol binario, deben hacerse una serie de operaciones que se encarguen de rearreglar a los nodos, de forma tal que la navegación del árbol muestre a los elementos ordenados en forma correcta (de menor a mayor en caso de navegación **inorder**). Estas operaciones requerirían no sólo un esfuerzo en programación, sino que el hecho de tener que reordenar los nodos del árbol consumiría tiempo de ejecución.

Eliminar físicamente los nodos de los árboles no permitiría la generación del reporte de elementos eliminados del acervo del sistema, ya que no habría ninguna forma de rastrear a los nodos eliminados.

Hasta este momento se han mencionado las políticas que se definieron para el manejo de los árboles del sistema. Sin embargo, recuérdese que cada árbol tiene asociada una lista doblemente encadenada; las operaciones de inserción, actualización y supresión en los árboles forzosamente afectan a las listas.

A continuación se muestran las políticas de inserción, actualización y supresión de las listas doblemente encadenadas del sistema:

> Cada vez que un nodo del árbol al cual se encuentra asociada la lista doblemente encadanada se añade o se suprime, la lista es creada nuevamente. Esto se debió a que se consideró que sería mucho más fácil y rápido que las listas se crearan nuevamente, a tener que actualizarlas añadiendo nuevos elementos implicando actualizaciones en los apuntadores de los elementos existentes.

5.5.3.- Manejo de archivos:

En este sistema, existen los siguientes archivos:

- Archivo para palabras clase y abreviaturas.
- Archivo para palabras entidad y abreviaturas.
- Archivo para nombres de diccionarios de datos.
- Archivo para nombres de ambientes de programación.
- Archivo para reglas de abreviación.
- Archivo para sustantivos que son excepciones a los gramemas caracteristicos.
- Archivo para adjetivos que son excepciones a los gramemas caracteristicos.

La caracteristica principal de estos archivos consiste en que son archivos históricos. Al momento de actualizar una palabra del acervo, ésta es copiada en un registro que se almacena en disco. Si dicha palabra ya existía en el archivo, entonces existirán 2 registros con información distinta para la misma palabra; sin embargo, el registro más nuevo (localizado más cerca del final del archivo) será el que contenga la información correcta. Lo mismo se aplica para el archivo de reglas de abreviación.

Las razones por las cuales se seleccionó que los archivos fuesen históricos, son las siguientes:

Se consideró que en futuras versiones del sistema sería conveniente tener la posibilidad de estudiar

la evolución del acervo de palabras válidas y sus correspondientes abreviaciones. De esta forma, teniendo archivos históricos, el sistema está potencialmente preparado para dar soporte a esta opción.

Debido a que el manejo de los datos se hace realmente a traves de las estructuras de datos dinámicas, en realidad no se requeria de un manejo muy complejo de archivos. El esfuerzo que hubiese requerido borrar físicamente registros de los archivos hubiese sido demasiado costoso en comparación al manejo realizado en memoria principal.

Por último, se consideró que en caso de que los archivos del sistema se perdieran (ya sea por que se borraron en forma intencional o accidental), el sistema deberia generar unos archivos con valores iniciales de palabras claves, asi como de nombres de diccionarios de datos, ambientes de programación y reglas de abreviación.

Cada árbol del sistema tiene asociado un archivo, asi que cada operación de inserción, actualización o supresión en los árboles implica una operación similar en sus archivos correspondientes.

5.5.4.- Manejo del teclado:

El teclado representa la vía en la que el usuario provee de información al sistema. Esta información de entrada puede verse de las siguientes formas:

- Cadenas de caracteres que forman palabras del texto de definición del dato a nombrar.
- Secuencias de caracteres (flechas, tecla Enter, o tecla ESC) usadas para seleccionar opciones en las ventanas de los menús del sistema.

Recuérdese que el sistema está basado en el uso de autómatas. Uno de los elementos de los autómatas son precisamente los caracteres o elementos terminales que forman parte del alfabeto reconocible por cada autómata. Debido a que en el sistema se usan distintos autómatas, fue necesario que el teclado fuese entonces controlado por distintas rutinas que fuesen capaces de aceptar sólo a aquellos caracteres (generados por las teclas oprimidas en el teclado) que pertenecieran al alfabeto reconocible por el autómata en ejecución. En otros casos, se requirió que las rutinas fuesen capaces de hacer ciertas operaciones de conversión, tal es el caso para el autómata que rige la aplicación del lenguaje de nomenclatura de datos. En este caso, la rutina que maneja al teclado se encarga de convertir a todas las letras mayúsculas en minúsculas, y las vocales acentuadas son convertidas a vocales no acentuadas; la barra espaciadora es interpretada como delimitador de palabras.

5.5.5.- Manejo de procesos de validación y seguridad del sistema:

Como medidas de seguridad en el sistema, se consideraron las siguientes:

Se definieron 3 tipos de usuarios, los cuales tendrían diferentes privilegios:

- Administrador de datos corporativo:
  Tiene acceso a todas las opciones y operaciones del sistema.
- Administrador de datos por área del negocio:
  Tiene acceso a todas las opciones y operaciones del sistema, excepto actualizar reglas de abreviación.
- Usuario normal:
  No puede actualizar al acervo del sistema.

Al momento de instalar al sistema, se le indican los nombres de los usuarios, sus claves de acceso así como su tipo de usuario.
Al momento de iniciar la ejecución del sistema, el usuario se identifica, dando su nombre y clave de acceso. El sistema verificará que concuerden con su registro de usuarios dados de alta al momento de instalar al sistema en el disco duro. Si el usuario es reconocido por el sistema, se le dará el acceso;

en caso contrario, se le negará. Para entrar al sistema, se disponen de 3 oportunidades antes de que la ejecución del sistema se aborte.

Asimismo, el sistema cuenta con rutinas de encripción de datos, por lo que los archivos usados para el control de usuarios se encuentran encriptados con el propósito de evitar que sean estudiados por personal no autorizado.

Debido a que las copias no autorizadas de software constituyen un problema muy delicado, se decidió que el sistema debería contar con una serie de mecanismos que eviten que sea copiado sin autorización. Estos mecanismos fueron instrumentados en forma de rutinas escritas en lenguaje ensamblador. Por motivos de la propia seguridad del sistema, no se describen aquí estos mecanismos.

La actualización del acervo del sistema requirió la consideración de una serie de mecanismos de validación, con el propósito de garantizar la calidad de la información almacenada. A continuación se muestran las premisas que sirvieron como base a estos mecanismos:

Sólo el personal autorizado con su correspondiente nivel, podrá hacer las operaciones que le son

permitidas. De esta forma, cada vez que se desee actualizar al acervo, el sistema verifica que el usuario cuente con la autorizacion requerida antes de llevar a cabo cualquier actualizacion.

Al dar de alta una nueva palabra clase, se verifica primero que:

- No se trate de un articulo o preposición.
- No se trate de una palabra almacenada como palabra entidad.

Al dar de alta una nueva palabra entidad, se verifica primero que:

- No se trate de un articulo o preposición.
- No se trate de una palabra almacenada como palabra clase.

Al momento de dar de alta o actualizar una abreviatura especial, se verifica antes que esta abreviatura no esté ya asociada con otra palabra del acervo del sistema. Si la abreviatura ya existe asociada a otra palabra, el sistema impide que se asocie a otra palabra. En tal caso, indica a cuál palabra ya está asociada dicha abreviatura, y pide al usuario que especifique una abreviatura distinta.

5.6.- Construcción del sistema:

5.6.1.- Carta estructurada del sistema.

El propósito de la carta estructurada es presentar en forma gráfica la relación existente entre las diferentes rutinas de un sistema. En esta representación gráfica, el sistema se asemeja a un árbol, en donde la raíz representa al programa principal y los nodos son las rutinas a las que se llaman unas a otras. Sin embargo, esta representación se vuelve difícil de hacer cuando existe una gran cantidad de rutinas. En el sistema de nomenclatura de datos existen más de 120 rutinas, algunas de las cuales se usan para el manejo de ventanas y que son invocadas por cualquier rutina. Representar esta situación a través de una carta estructurada sería muy difícil y poco práctico. Por esta razón, se decidió sustituir a la carta estructurada por una matriz que presente las relaciones entre las rutinas del sistema. Así, si el elemento M[i,j] tiene una "X" significa que la rutina del i-ésimo renglón utiliza a la rutina de la j-ésima columna. Las rutinas se han agrupado por "unidades". Una "unidad" en turbo Pascal representa un conjunto de rutinas que se almacenan en un sólo archivo y que pueden compilarse para formar una biblioteca de rutinas. El nombre de cada "unidad" se ha escrito en mayúsculas, mientras que los nombres de las rutinas aparecen en minúsculas. Por ser muy grande la matriz, se dividió en 25 secciones o submatrices. En éstas, aparece una referencia gráfica de la posicion de la submatriz con respecto a la matriz completa.

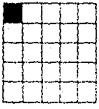
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PGM. PRINCIPAL	arma_semejantes	lee_dato	une_texto	semantica	autom_error	ini_errores	autom_sintaxis	inicia_sintaxis	inicia_listas	limpia_pila	MORFEMAS	ini_morf	autom_morf	ver_sustantivo	ver_adjetivo	MENUS	autom_menus	ini_menus	corrige_texto	ABRCION	es_plural	quita_plural	prep_abrevia	autom_abrevia	ini_a_abrevia	AUTO_REP	sig_tecla	ini_aux_rep	guarda_reglas_rep	aut_aux_rep	ini_rep	ini_orden_imp	ini_tipo_imp	guarda_orden_rep	autom_reps
PGM. PRINCIPAL							x	xx	x	x		+	x				+		x		+				x	x	+									
arma_semejantes	x																	x																		
lee_dato																																				
une_texto																																				
semantica															xx																					
autom_error	x		x																																	
ini_errores																																				
autom_sintaxis		x		xx	x													x																		
inicia_sintaxis																																				
inicia_listas																																				
limpia_pila																																				
MORFEMAS																					+															
ini_morf																																				
autom_morf																																				
ver_sustantivo														x									x													
ver_adjetivo														x									x													
MENUS																	+		+								+								x
autom_menus																		x					x													
ini_menus																																				
corrige_texto																																				
ABRCION																											+									
es_plural																																				
quita_plural																																				
prep_abrevia																																				
autom_abrevia																							x													
ini_a_abrevia																																				
AUTO_REP																																				
sig_tecla																																				
ini_aux_rep																																				
guarda_reglas_rep																																				
aut_aux_rep																																				
ini_rep																																				
ini_orden_imp																																				
ini_tipo_imp																																				
guarda_orden_rep																																				
autom_reps																												x	x	x	x	x				

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

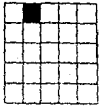
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	MAN_REP	imprime_arbol	imprime_borrados	cuenta_borrados	imp_abrev	imp_clase	imp_entidad	imp_ddds	imp_amb_prog	imp_reglas	imp_el_clase	imp_el_entidad	imp_el_ddds	imp_el_amb_prog	imp_nom_prim	imp_nom_prog	MAN_TMP	limpia_buffer	respalda_buffer	scroll_texto	imp_reloj	pon_header	pon_fin_pagina	guarda_buffer	formatea	REGLAS	display_regla	autom_reglas	inicia_reglas	recibe_numero
PGM_PRINCIPAL	+																+									+			x	
arma_semejantes																														
lee_dato																														
une_texto																														
semantica																														
autom_error																														
ini_errores																														
autom_sintaxis																														
inicia_sintaxis																														
inicia_listas																														
limpia_pila																														
MORFEMAS																														
ini_morf																														
autom_morf																														
ver_sustantivo																														
ver_adjetivo																														
MENUS																														
autom_menus																														
ini_menus																														
corrige_texto																														
ABRCION	+																+									+				
es_plural																														
quita_plural																														
prep_abrevia																														
autom_abrevia															x	x				x	x	x					x			
ini_a_abrevia																														
AUTO_REP	+																+													
sig_tecla																														
ini_aux_rep																														
guarda_reglas_rep																														
aut_aux_rep																														
ini_rep																														
ini_orden_imp																														
ini_tipo_imp																														
guarda_orden_rep																														
autom_reps	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x				x	x		x							

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

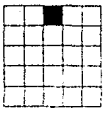
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	SEGURIDAD	autentificacion	autorizado	verificado	mismo_disco	INICIOS	ini_entidad	ini_ddds	ini_amb_prog	ini_clase	ini_arboles	MAN_ARCH	id_unidad_drive	nm_unidad_drive	anade_palabra	trae_palabra	anade_regla	trae_regla	SCANNER	sin_cursor	con_cursor	oprime_esc_int	letra_a_m	espera_o_f	espera_b_o_f	cualquier_tecla	scan_editor	scan_numero	scan_caresp
PGM. PRINCIPAL	+	x			x	+					x	+						x	+	x	x								
arma_semejantes																											x		
lee_dato																													
une_texto																													
semantica																													
autom_error																							x		x				
ini_errores																													
autom_sintaxis																											x		
inicia_sintaxis																													
inicia_listas																													
limpia_pila																													
MORFEMAS	+					+						+																	
ini_morf																													
autom_morf																													
ver_sustantivo			x												x														
ver_adjetivo			x												x														
MENUS	+					+						+							+										
autom_menus			x	x																					x	x			
ini_menus																													
corrige_texto																													
ABRCION												+							+										
es_plural																													
quita_plural																													
prep_abrevia																													
autom_abrevia																		x									x		
ini_a_abrevia																													
AUTO_REP												+							+										
sig_tecla																													
ini_aux_rep																													
guarda_reglas_rep																													
aut_aux_rep																											x		
ini_rep																													
ini_orden_imp																													
ini_tipo_imp																													
guarda_orden_rep																													
autom_reps																		x									x		

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

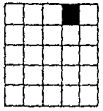
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PILAS	push_stack	enciende_mensaje	apaga_mensaje	ARBOLES	ins_arbol	consulta_arbol	busca_arbol	busca_modificador	pon_en_lista	arma_lista	recorre_lista	inicia_cabeza	ven_listas	semejanza	crea_arb_abrev	cuenta_arbol	abrev_repetida	borra_arbol	borra_lista	arma_arb_abrev	TIEMPOS	tm_inicial	reloj	tiempo	MAN_I_VTA	construye_ventana
PGM_PRINCIPAL	+				+																	+	x	x	x	+	x
arma_semejantes						x																					
lee_dato																											
une_texto		x																									
semantica																											
autom_error											x			x					x	x							x
ini_errores																											
autom_sintaxis																											
inicia_sintaxis																											
inicia_listas																											
limpia_pila																											
MORFEMAS					+																					+	
ini_morf																											
autom_morf																											
ver_sustantivo						x		x																			x
ver_adjetivo						x		x																			x
MENUS	+				+																					+	
autom_menus						x		x		x		x								x	x						x
ini_menus																											
corrige_texto			x	x																							
ABRCION	+				+																	+				+	
es_plural																											
quita_plural																											
prep_abrevia																											
autom_abrevia							x	x				x											x				x
ini_a_abrevia																											
AUTO_REP					+																					+	
sig_tecla																											
ini_aux_rep																											
guarda_reglas_rep																											
aut_aux_rep								x				x															x
ini_rep																											
ini_orden_imp																											
ini_tipo_imp																											
guarda_orden_rep																											
autom_reps																											x

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

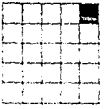
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	D_VTA_1	inicio_1	ini_1_ventanas	gen_vta	MAIN_VTA	grafica_pantalla	quita_ventana	quita_todas	pon_ven_mensaje	pon_cad_mensaje	pon_mensaje	escribe_mensaje	scrol_mensaje	desactiva_mensaje	activa_mensaje	desact_ren_vta	act_ren_vta	ini_ap_vtas
PGM. PRINCIPAL	+	x	x		+	x	x	x										x
arma_semejantes																		
lee_dato												x						
une_texto																		
semantica																		
autom_error								x	x	x								
ini_errores																		
autom_sintaxis												x			x			
inicia_sintaxis																		
inicia_listas																		
limpia_pila																		
MORFEMAS					+													
ini_morf																		
autom_morf																		
ver_sustantivo							x		x	x								
ver_adjetivo							x		x	x								
MENUS					+													
autom_menus							x	x	x	x		x					x	
ini_menus																		
corrige_texto												x			x			
ABRCION					+													
es_plural																		
quita_plural													x					
prep_abrevia																		
autom_abrevia							x	x	x			x	x		x		x	
ini_a_abrevia																		
AUTO_REP					+													
sig_tecla																		
ini_aux_rep																		
guarda_reglas_rep																		
aut_aux_rep							x										x	
ini_rep																		
ini_orden_imp																		
ini_tipo_imp																		
guarda_orden_rep																		
autom_reps							x	x		x							x	

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

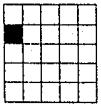
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PGM_PRINCIPAL	arma_semejantes	lee_dato	une_texto	semantica	autom_error	ini_errores	autom_sintaxis	inicia_sintaxis	inicia_listas	limpia_pila	MORFEMAS	ini_morf	autom_morf	ver_sustantivo	ver_adjetivo	MENUS	autom_menus	ini_menus	corrige_texto	ABERCION	es_plural	cuta_plural	prep_abrevia	autom_abrevia	ini_a_abrevia	AUTO_REP	sig_tecla	ini_aux_rep	guarda_reglas_rep	aut_aux_rep	ini_rep	ini_orden_imp	ini_tipo_imp	guarda_orden_rep	autom_reps
MAN_REP																																				
imprime_arbol																																				
imprime_borrados																																				
cuenta_borrados																																				
imp_abrev																																				
imp_clase																																				
imp_entidad																																				
imp_ddds																																				
imp_amb_prog																																				
imp_reglas																																				
imp_eli_clase																																				
imp_eli_entidad																																				
imp_eli_ddds																																				
imp_eli_amb_prog																																				
imp_nom_prim																																				
imp_nom_prog																																				
MAN_IMP																																				
limpia_buffer																																				
respalda_buffer																																				
scroll_texto																																				
imp_reloj																																				
pon_header																																				
pon_fin_pagina																																				
guarda_buffer																																				
formatea																																				
REGLAS																																				
display_regla																																				
autom_reglas																																				
inicia_reglas																																				
recibe_numero																																				

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

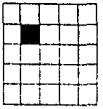
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	MAN_REP	imprime_arbol	imprime_borrados	cuenta_borrados	imp_abrev	imp_clase	imp_entidad	imp_ddds	imp_amb_prog	imp_reglas	imp_eli_clase	imp_eli_entidad	imp_eli_ddds	imp_eli_amb_prog	imp_nom_prim	imp_nom_prog	MAN_IMP	limpia_buffer	respalda_buffer	scroll_texto	imp_reloj	pon_header	pon_fin_pagina	guarda_buffer	formatea	REGLAS	display_regla	autom_reglas	inicia_reglas	recibe_numero
MAN_REP																	+													
imprime_arbol		x																		x					x					
imprime_borrados			x																	x					x					
cuenta_borrados				x																										
imp_abrev		x																		x		x			x					
imp_clase		x																		x		x			x					
imp_entidad		x																		x		x			x					
imp_ddds		x																		x		x			x					
imp_amb_prog		x																		x		x			x					
imp_reglas																				x		x								
imp_eli_clase			x	x																x		x			x					
imp_eli_entidad			x	x																x		x			x					
imp_eli_ddds			x	x																x		x			x					
imp_eli_amb_prog			x	x																x		x			x					
imp_nom_prim																				x										
imp_nom_prog																				x										
MAN_IMP																														
limpia_buffer																			x											
respalda_buffer																				x										
scroll_texto																					x									
imp_reloj																														
pon_header																			x											
pon_fin_pagina																				x										
guarda_buffer																				x										
formatea																														
REGLAS																														
display_regla																														
autom_reglas																											x		x	
inicia_reglas																														
recibe_numero																														

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

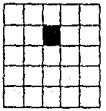
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	SEGURIDAD	autentificacion	autorizado	verificado	mismo_disco	INICIOS	ini_entidad	ini_ddds	ini_amb_prog	ini_clase	ini_arboles	MAN_ARCH	id_unidad_drive	nm_unidad_drive	anade_palabra	trae_palabra	anade_regla	trae_regla	SCANNER	scn_cursor	con_cursor	oprime_esc_init	letra_a_m	espera_o_f	espera_b_o_f	cualquier_tecla	scan_editor	scan_numro	scan_caresp
MAN_REP												..																	
imprime_arbol																													
imprime_borrados																													
cuenta_borrados																													
imp_abrev																													
imp_clase																													
imp_entidad																													
imp_ddds																													
imp_amb_prog																													
imp_reglas																													
imp_eli_clase																													
imp_eli_entidad																													
imp_eli_ddds																													
imp_eli_amb_prog																													
imp_nom_prim																													
imp_nom_prog																													
MAN_IMP																													
limpia_buffer																													
respalda_buffer																													
scroll_texto																													
imp_reloj																													
pon_header																													
pon_fin_pagina																													
guarda_buffer																													
formatea																													
REGLAS	+											+						+											
display_regla																													
autom_reglas			x	x												x	x								x	..		
inicia_reglas																													
recibe_numero			x																							x	..		

Loc. matriz general:



SIMBOLOGIA: X .. Llama a la rutina
+ .. Requiere a la unidad

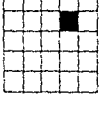
Tesis. AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PILAS	push_stack	enciende_mensaje	apaga_mensaje	ARBOLES	ins_arbol	consulta_arbol	busca_arbol	busca_modificador	pon_en_lista	arma_lista	recorre_lista	inicia_cabeza	ven_listas	semejanza	crea_arb_abrev	cuenta_arbol	abrev_repetida	borra_arbol	borra_lista	arma_arb_abrev	TIEMPOS	tm_inicial	reloj	tiempo	MAN_1_VTA	construye_ventana
MAN_REP																											
imprime_arbol																											
imprime_borrados																											
cuenta_borrados																											
imp_abrev																											
imp_clase																											
imp_entidad																											
imp_ddds																											
imp_amb_prog																											
imp_reglas																											
imp_eli_clase																											
imp_eli_entidad																											
imp_eli_ddds																											
imp_eli_amb_prog																											
imp_nom_prim																											
imp_nom_prog																											
MAN_IMP																											
limpia_buffer																											
respalda_buffer																											
scroll_texto																											
imp_reloj																											
pon_header																											
pon_fin_pagina																											
guarda_buffer																											
formatea																											
REGLAS					+																				+		
display_regla																										x	
autom_reglas						x		x		x		x								x						x	
inicia_reglas																											
recibe_numero																											

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la *rutina*
+ -- Requiere a la unidad

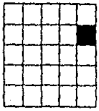
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	D_VTA_1	inicio_1	ini_1_ventanas	gen_vta	MAN_VTA	grafica_pantalla	quita_ventana	quita_todas	pon_ven_mensaje	pon_cad_mensaje	pon_mensaje	escribe_mensaje	scroll_mensaje	desactiva_mensaje	activa_mensaje	desact_ren_vta	act_ren_vta	ini_ap_vtas
MAN_REP																		
imprime_arbol																		
imprime_borrados																		
cuenta_borrados																		
imp_abrev																		
imp_clase																		
imp_entidad																		
imp_ddds																		
imp_amb_prog																		
imp_reglas																		
imp_eli_clase																		
imp_eli_entidad																		
imp_eli_ddds																		
imp_eli_amb_prog																		
imp_nom_prim																		
imp_nom_prog																		
MAN_IMP																		
limpia_buffer																		
respalda_buffer																		
scroll_texto																		
imp_reloj																		
pon_header																		
pon_fin_pagina																		
guarda_buffer																		
formatea																		
REGLAS																		
display_regla									x									
autom_reglas							x									x	x	
inicia_reglas																		
recibe_numero																		

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

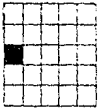
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PGM. PRINCIPAL	arma_semejantes	lee_dato	une_texto	semantica	autom_error	mi_errores	_autom_sintaxis	inicia_sintaxis	inicia_listas	limpia_pila	MORFEMAS	ini_morf	autom_morf	ver_sustantivo	ver_adjetivo	MENUS	autom_menus	ini_menus	cambia_texto	ABRICION	es_plural	quita_plural	prep_abrevia	autom_abrevia	ini_a_abrevia	AUTO_REP	sig_tecla	ini_aux_rep	guarda_reglas_rep	aut_aux_rep	ini_rep	mi_orden_imp	ini_tipo_imp	guarda_orden_rep	autom_reps
SEGURIDAD																																					
autentificacion																																					
autorizado																																					
verificado																																					
mismo_disco																																					
INICIOS																																					
ini_entidad																																					
ini_ddds																																					
ini_amb_prog																																					
ini_clase																																					
ini_arboles																																					
MAN_ARCH																																					
id_unidad_drive																																					
nm_unidad_drive																																					
anade_palabra																																					
trae_palabra																																					
anade_regla																																					
trae_regla																																					
SCANNER																																					
sin_cursor																																					
con_cursor																																					
oprime_esc_int																																					
letra_a_m																																					
espera_o_f																																					
espera_b_o_f																																					
cualquier_tecla																																					
scan_editor																																					
scan_numro																																					
scan_caresp																																					

Loc. matriz general:



SIMBOLOGIA X -- Llama a la *rutina*

\+ -- Requiere a la unidad

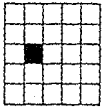
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	MAN_REP	imprime_arbol	imprime_borrados	cuenta_borrados	imp_abrev	imp_clase	imp_entidad	imp_ddds	imp_amb_prog	imp_reglas	imp_eli_clase	imp_eli_entidad	imp_eli_ddds	imp_eli_amb_prog	imp_nom_prim	imp_nom_prog	MAN_IMP	limpia_buffer	respalda_buffer	scroll_texto	imp_reloj	pon_header	pon_fin_pagina	guarda_buffer	formatea	REGLAS	display_regla	autom_reglas	inicia_reglas	recibe_numero
SEGURIDAD																														
autentificacion																														
autorizado																														
verificado																														
mismo_disco																														
INICIOS																														
ini_entidad																														
ini_ddds																														
ini_amb_prog																														
ini_clase																														
ini_arboles																														
MAN_ARCH																														
id_unidad_drive																														
nm_unidad_drive																														
anade_palabra																														
trae_palabra																														
anade_regla																														
trae_regla																														
SCANNER																														
sin_cursor																														
con_cursor																														
oprime_esc_int																														
letra_a_m																														
espera_o_f																														
espera_b_o_f																														
cualquier_tecla																														
scan_editor																														
scan_numro																														
scan_caresp																														

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

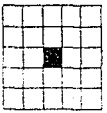
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G Vázquez Nava

Rutinas que hacen llamadas a otras rutinas \ Rutinas que son llamadas por otras rutinas	SEGURIDAD	autentificacion	autorizado	verificado	mismo_disco	INICIOS	ini_entidad	ini_ddds	ini_amb_prog	ini_clase	ini_arboles	MAN_ARCH	id_unidad_drive	nm_unidad_drive	anade_palabra	trae_palabra	anade_regla	trae_regla	SCANNER	sin_cursor	con_cursor	oprime_esc_int	letra_a_m	espera_b_o_f	cualquier_tecla	scan_editor	scan_numro	scan_caresp
SEGURIDAD																			+									
autentificacion																									x			
autorizado																												
verificado																												
mismo_disco													x															
INICIOS												+																
ini_entidad															x													
ini_ddds															x													
ini_amb_prog															x													
ini_clase															x													
ini_arboles							x	x	x	x	x					x												
MAN_ARCH																												
id_unidad_drive																												
nm_unidad_drive													x															
anade_palabra																												
trae_palabra																												
anade_regla																												
trae_regla																	x											
SCANNER																												
sin_cursor																												
con_cursor																												
oprime_esc_int																												
letra_a_m																												
espera_o_f																												
espera_b_o_f																												
cualquier_tecla																												
scan_editor																				x	x							
scan_numro																				x	x							
scan_caresp																				x	x							

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

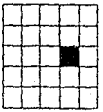
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PILAS	push_stack	encuende_mensaje	apaga_mensaje	ARBOLES	ins_arbol	consulta_arbol	busca_arbol	busca_modificador	pon_en_lista	arma_lista	recorre_lista	inicia_cabeza	ven_listas	semejanza	crea_arb_abrev	cuenta_arbol	abrev_repetida	borra_arbol	borra_lista	arma_arb_abrev	TIEMPOS	tm_inicial	reloj	tiempo	MAN_1_VTA	construye_ventana
SEGURIDAD					+																					+	
autentificacion							x	x																			x
autorizado									x																		x
verificado																											x
mismo_disco							x	x																			
INICIOS					+																						
ini_entidad						x																					
ini_ddds						x																					
ini_amb_prog						x																					
ini_clase						x																					
ini_arboles						x					x					x											
MAN_ARCH																											
id_unidad_drive																											
nm_unidad_drive																											
anade_palabra																											
trae_palabra																											
anade_regla																											
trae_regla																											
SCANNER																											
sin_cursor																											
con_cursor																											
oprime_esc_int																											
letra_a_m																											
espera_o_f																											
espera_b_o_f																											
cualquier_tecla																											
scan_editor																											
scan_numro																											
scan_caresp																											

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

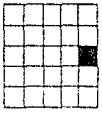
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	D_VTA_1	inicio_1	ini_1_ventanas	gen_vta	MAN_VTA	grafica_pantalla	quita_ventana	quita_todas	pon_ven_mensaje	pon_cad_mensaje	pon_mensaje	escribe_mensaje	scrol_mensaje	desactiva_mensaje	activa_mensaje	desact_ren_vta	act_ren_vta	ini_ap_vtas
SEGURIDAD					+													
autentificacion							x	x										
autorizado							x											
verificado							x											
mismo_disco																		
INICIOS																		
ini_entidad																		
ini_ddds																		
ini_amb_prog																		
ini_clase																		
ini_arbolos																		
MAN_ARCH																		
id_unidad_drive																		
nm_unidad_drive																		
anade_palabra																		
trae_palabra																		
anade_regla																		
trae_regla																		
SCANNER																		
sin_cursor																		
con_cursor																		
oprime_esc_int																		
letra_a_m																		
espera_o_f																		
espera_b_o_f																		
cualquier_tecla																		
scan_editor																		
scan_numro																		
scan_caresp																		

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

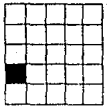
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PGM_PRINCIPAL	arma_semejantes	lee_dato	une_texto	semántica	autom_error	ini_errores	autom_sintaxis	inicia_sintaxis	inicia_listas	limpia_pila	MORFEMAS	ini_morf	autom_morf	ver_sustantivo	ver_adjetivo	MENUS	autom_menus	ini_menus	corrige_texto	ABRCION	es_plural	quita_plural	prep_abrevia	autom_abrevia	ini_a_abrevia	AUTO_REP	sig_tecla	ini_aux_rep	guarda_reglas_rep	aut_aux_rep	ini_rep	ini_orden_imp	ini_tipo_imp	guarda_orden_rep	autom_reps
PILAS																																					
push_stack																																					
enciende_mensaje																																					
apaga_mensaje																																					
ARBOLES																																					
ins_arbol																																					
consulta_arbol																																					
busca_arbol																																					
busca_modificador																																					
pon_en_lista																																					
arma_lista																																					
recorre_lista																																					
inicia_cabeza																																					
ven_listas																																					
semejanza																																					
crea_arb_abrev																																					
cuenta_arbol																																					
abrev_repetida																																					
borra_arbol																																					
borra_lista																																					
arma_arb_abrev																																					
TIEMPOS																																					
tim_inicial																																					
reloj																																					
tiempo																																					
MAN_1_VTA																																					
construye_ventana																																					

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

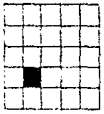
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	MAN_REP	imprime_arbol	imprime_borrados	cuenta_borrados	imp_abrev	imp_clase	imp_entidad	imp_ddds	imp_amb_prog	imp_reglas	imp_el_clase	imp_el_entidad	imp_el_ddds	imp_el_amb_prog	imp_nom_prim	imp_nom_prog	MAN_TMP	limpia_buffer	respalda_buffer	scroll_texto	imp_reloj	pon_header	pon_fin_pagina	guarda_buffer	formatea	REGLAS	display_regla	autom_reglas	inicia_reglas	redbe_numero
PILAS																														
push_stack																														
enciende_mensaje																														
apaga_mensaje																														
ARBOLES																														
ins_arbol																														
consulta_arbol																														
busca_arbol																														
busca_modificador																														
pon_en_lista																														
arma_lista																														
recorre_lista																														
inicia_cabeza																														
ven_listas																														
semejanza																														
crea_arb_abrev																														
cuenta_arbol																														
abrev_repetida																														
borra_arbol																														
borra_lista																														
arma_arb_abrev																														
TIEMPOS																														
tm_inicial																														
reloj																														
tiempo																														
MAN_1_VTA																														
construye_ventana																														

Loc. matriz general:



SIMBOLOGIA X -- Llama a la rutina
+ -- Requiere a la unidad

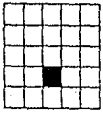
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	SEGURIDAD	autentificacion	autorizado	verificado	mismo_disco	INICIOS	ini_entidad	ini_cdds	ini_amb_prog	ini_clase	ini_arboles	MAN_ARCH	id_unidad_drive	nm_unidad_drive	anade_palabra	trae_palabra	anade_regla	trae_regla	SCANNER	sin_cursor	con_cursor	oprime_esc_int	letra_a_m	espera_b_o_f	cualquier_tecla	scan_editor	scan_numro	scan_caresp
PILAS																												
push_stack																												
enciende_mensaje																												
apaga_mensaje																												
ARBOLES																												
ins_arbol																												
consulta_arbol																												
busca_arbol																												
busca_modificador																												
pon_en_lista																												
arma_lista																												
recorre_lista																												
inicia_cabeza																												
ven_listas																												
semejanza																												
crea_arb_abrev																												
cuenta_arbol																												
abrev_repetida																												
borra_arbol																												
borra_lista																												
arma_arb_abrev																												
TIEMPOS																												
tim_inicial																												
reloj																												
tiempo																												
MAN_1_VTA																												
construye_ventana																												

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

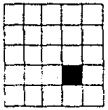
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PILAS	push_stack	enciende_mensaje	apaga_mensaje	ARBOLES	ins_arbol	consulta_arbol	busca_arbol	busca_modificador	pon_en_lista	arma_lista	recorre_lista	inicia_cabeza	ven_listas	semejanza	crea_arb_abrev	cuenta_arbol	abrev_repetida	borra_arbol	borra_lista	arma_arb_abrev	TIEMPOS	tim_inicial	reloj	tiempo	MAN_1_VTA	construye_ventana
PILAS																											
push_stack																											
enciende_mensaje																											
apaga_mensaje																											
ARBOLES																									+		
ins_arbol						x																					
consulta_arbol							x																				
busca_arbol								x																			
busca_modificador									x																		
pon_en_lista																											
arma_lista											x	x															
recorre_lista													x														
inicia_cabeza																											
ven_listas														x												x	
semejanza																											
crea_arb_abrev						x										x											
cuenta_arbol																	x										
abrev_repetida							x	x																		x	
borra_arbol																			x								
borra_lista																				x							
arma_arb_abrev																x			x								
TIEMPOS																											
tim_inicial																											
reloj																											
tiempo																											
MAN_1_VTA																											
construye_ventana																											

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

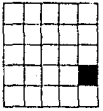
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que hacen llamadas a otras rutinas \ Rutinas que son llamadas por otras rutinas	D_VTA_1	inicio_1	ini_1_ventanas	gen_vta	MAN_VTA	grafica_pantalla	quita_ventana	quita_todas	pon_ven_mensaje	pon_cad_mensaje	pon_mensaje	escribe_mensaje	scroll_mensaje	desactiva_mensaje	activa_mensaje	desact_ren_vta	act_ren_vta	ini_ap_vtas
PILAS																		
push_stack																		
enciende_mensaje																		
apaga_mensaje																		
ARBOLES					+													
ins_arbol																		
consulta_arbol																		
busca_arbol																		
busca_modificador																		
pon_en_lista																		
arma_lista																		
recorre_lista																		
inicia_cabeza																		
ven_listas																		
semejanza																		
crea_arb_abrev																		
cuenta_arbol																		
abrev_repetida							x											
borra_arbol																		
borra_lista																		
arma_arb_abrev																		
TIEMPOS																		
tim_inicial																		
reloj																		
tiempo																		
MAN_1_VTA																		
construye_ventana																		

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

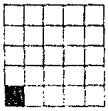
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PGM_PRINCIPAL	arma_semejantes	lee_dato	une_texto	semantica	autom_error	ini_errores	_autom_sintaxis	inicia_sintaxis	inicia_listas	limpia_pila	MORFEMAS	ini_morf	autom_morf	ver_sustantivo	ver_adjetivo	MENUS	autom_menus	ini_menus	corrige_texto	ABRCION	es_plural	quita_plural	prep_abrevia	autom_abrevia	ini_a_abrevia	AUTO_REP	sig_tecla	ini_aux_rep	guarda_reglas_rep	aut_aux_rep	ini_rep	ini_orden_imp	ini_tipo_imp	guarda_orden_rep	autom_reps
D_VTA_1																																				
inicio_1																																				
ini_1_ventanas																																				
gen_vta																																				
MAN_VTA																																				
grafica_pantalla																																				
quita_ventana																																				
quita_todas																																				
pon_ven_mensaje																																				
pon_cad_mensaje																																				
pon_mensaje																																				
escribe_mensaje																																				
scroll_mensaje																																				
desactiva_mensaje																																				
activa_mensaje																																				
desact_ren_vta																																				
act_ren_vta																																				
ini_ap_vtas																																				

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

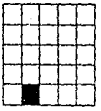
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	MAN_REP	imprime_arbol	imprime_borrados	cuenta_borrados	imp_abrev	imp_clase	imp_entidad	imp_ddds	imp_amb_prog	imp_reglas	imp_el_clase	imp_el_entidad	imp_el_ddds	imp_el_amb_prog	imp_nom_prim	imp_nom_prog	MAN_IMP	limpia_buffer	respalda_buffer	scroll_texto	imp_reloj	pon_header	pon_fin_pagina	guarda_buffer	formatea	REGLAS	display_regla	autom_reglas	inicia_reglas	recibe_numero
D_VTA_1 inicio_1 ini_1_ventanas gen_vta																														
MAN_VTA grafica_pantalla quita_ventana quita_todas pon_ven_mensaje pon_cad_mensaje pon_mensaje escribe_mensaje scroll_mensaje desactiva_mensaje activa_mensaje desact_ren_vta act_ren_vta ini_ap_vtas																														

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

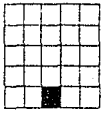
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J. G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	SEGURIDAD	autentificacion	autorizado	verificado	mismo_disco	INICIOS	ini_entidad	ini_ddds	ini_amb_prog	ini_clase	ini_arboles	MAN_ARCH	id_unidad_drive	nm_unidad_drive	anade_palabra	trae_palabra	anade_regla	trae_regla	SCANNER	scn_cursor	con_cursor	oprime_esc_int	letra_a_m	espera_o_f	espera_b_o_f	cualquier_tecla	scan_editor	scan_numro	scan_caresp
D_VTA_1																													
inicio_1																													
ini_1_ventanas																													
gen_vta																													
MAN_VTA																													
grafica_pantalla																													
quita_ventana																													
quita_todas																													
pon_ven_mensaje																													
pon_cad_mensaje																													
pon_mensaje																													
escribe_mensaje																													
scroll_mensaje																													
desactiva_mensaje																													
activa_mensaje																													
desact_ren_vta																													
act_ren_vta																													
ini_ap_vtas																													

Loc. matriz general:



SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

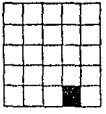
Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	PILAS	push_stack	enciende_mensaje	apaga_mensaje	ARBOLES	ins_arbol	consulta_arbol	busca_arbol	busca_modificador	pon_en_lista	arma_lista	recorre_lista	inicia_cabeza	ven_listas	semejanza	crea_arb_abrev	cuenta_arbol	abrev_repetida	borra_arbol	borra_lista	arma_arb_abrev	TIEMPOS	tm_inicial	reloj	tiempo	MAN_1_VTA	construye_ventana
D_VTA_1 inicio_1 ini_1_ventanas gen_vta																											
MAN_VTA grafica_pantalla quita_ventana quita_todas pon_ven_mensaje pon_cad_mensaje pon_mensaje escribe_mensaje scroll_mensaje desactiva_mensaje activa_mensaje desact_ren_vta act_ren_vta ini_ap_vtas																											

Loc. matriz general:



SIMBOLOGIA: X – Llama a la rutina
+ – Requiere a la unidad

Tesis: AUTOMATIZACION DE ESTANDARES COMO UNA DE LAS FUNCIONES DE LA ADMINISTRACION DE DATOS

Matriz para relacionar rutinas del sistema de nomenclatura de datos

Alejandro J.G. Vázquez Nava

Rutinas que son llamadas por otras rutinas / Rutinas que hacen llamadas a otras rutinas	D_VTA_1	inicio_1	ini_1_ventanas	gen_vta	MAN_VTA	grafica_pantalla	quita_ventana	quita_todas	pon_ven_mensaje	pon_cad_mensaje	pon_mensaje	escribe_mensaje	scroll_mensaje	desactiva_mensaje	activa_mensaje	desact_ren_vta	act_ren_vta	ini_ap_vtas
D_VTA_1					+													
inicio_1											x							
ini_1_ventanas																		
gen_vta									x									
MAN_VTA																		
grafica_pantalla																		
quita_ventana																		
quita_todas							x											
pon_ven_mensaje																		
pon_cad_mensaje									x									
pon_mensaje																		
escribe_mensaje																		
scroll_mensaje																		
desactiva_mensaje																		
activa_mensaje															x			
desact_ren_vta																		
act_ren_vta																x		
ini_ap_vtas																		

Loc. matriz general:

SIMBOLOGIA: X -- Llama a la rutina
+ -- Requiere a la unidad

5.6.2.- Descripción de las principales rutinas del sistema:

A continuación se muestra la descripción de las principales rutinas del sistema. En esta descripción, se está siguiendo el siguiente formato, el cual se explica a continuación:

Nombre de la unidad : En turbo Pascal, es posible tener varias rutinas almacenadas en un sólo archivo. Asimismo, en este archivo es posible tener varias variables que se desea sean consideradas por el compilador como variables globales para el programa principal que usa a dicho archivo. Este archivo recibe el nombre de **unidad.**

Unidades requeridas : Se refiere a las unidades que se requieren incluir en la presente unidad, para que ésta pueda ser compilada exitosamente.

Descripción general : Indica el propósito global de las rutinas que forman parte de la unidad.

Rutinas de esta unidad: Se refiere a las rutinas almacenadas en la presente unidad.

Procedimiento/ función : Se refiere al nombre del procedimiento

o funcion que se describe. En caso de tratarse de una función, se incluye el tipo de ésta.

Parámetros : En el caso de que la rutina requiera de parámetros, en esta sección se describen los parametros utilizados.

Descripción : En esta sección, se explican las acciones que realiza la rutina que se describe.

Ejemplo : Esta seccion muestra cómo podria invocarse a la rutina desde un programa que utilizase a esta unidad.

A continuación se describen las unidades y sus correspondientes rutinas, siguiendo el formato arriba descrito:

Nombre de la unidad : **man_vta**

Unidades requeridas : crt

Descripción general :
Las rutinas de esta unidad tienen como propósito formar una infraestructura que permita la definición y manejo de las ventanas del sistema.

Rutinas de esta unidad:

Procedimiento : grafica_pantalla
Parámetros : número_pantalla:integer
Descripción:
Esta rutina se encarga de almacenar en la memoria de video a la pantalla cuyo número de

identificación se especifica en el parámetro "número_pantalla". Asimismo, se encarga de hacer el despliegue de dicha pantalla en el monitor de la computadora.

Ejemplo : grafica_pantalla(1);

Procedimiento : quita_ventana
Descripción:
Esta rutina se encarga de quitar la ventana que se encuentra hasta el tope de la pila de ventanas del sistema.
En el monitor desaparece dicha ventana, y aparece lo que hubiera estado abajo de ella.

Procedimiento : quita_todas
Descripción:
Esta rutina se encarga de quitar todas las ventanas que se encuentren en la pila de ventanas del sistema.
En el monitor desaparecen todas las ventanas .

Procedimiento : pon_ven_mensaje
Parámetros : número_ventana : integer
mensaje : cadena
color : byte
renglón,
columna : integer

Descripción:
Esta rutina se utiliza para la definición de ventanas. Permite colocar mensajes en cada renglón de la ventana cuya identificación se especifica a través del parámetro "número_ventana". A través de "mensaje" se indica el mensaje a definir en la ventana y con qué color de fondo. Los parámetros "renglón" y "columna" indican a esta rutina el número de renglón de la ventana y la columna en donde deberá aparecer el mensaje.

Ejemplo :
pon_ven_mensaje(5,'Nombre del autor',$4F,3,3);
esta instrucción coloca al mensaje "Nombre del autor" en fondo rojo y letras blancas, en el renglón #3, columna #3 de la ventana #5.

Procedimiento : pon_cad_mensaje
Parámetros : número_ventana : integer
mensaje : cadena
renglón : integer

Descripción:
Esta rutina se utiliza para colocar un

mensaje en un renglón de determinada ventana, de forma tal que quede en el centro.

Ejemplo :
pon_cad_mensaje(3,dato_leido,4);
Esta instrucción coloca al contenido de la variable "dato_leido" al centro del renglón #4 de la ventana #3.

Procedimiento : pon_mensaje
Parámetros : número_pantalla: integer
mensaje : cadena
color : byte
renglon,
columna : integer

Descripción:
Esta rutina sirve para definir pantallas. Se utiliza para colocar un mensaje con cierto color, en el renglón y columna de la pantalla especificada a través de los parámetros indicados arriba.

Ejemplo :
pon_mensaje(1,'Mensaje del sistema',$4F,5,12);
La instrucción anterior coloca al mensaje "Mensaje del sistema" con color de fondo rojo, letras blancas, en el renglón #5, columna #12, de la pantalla #1.

Procedimiento : escribe_mensaje
Parámetros : mensaje : cadena
renglón,
columna : integer

Descripción:
Esta rutina permite añadir mensajes a pantallas que ya están definidas y que están siendo desplegadas a través de la memoria de video. El mensaje aparecerá en el renglón y columna especificadas, en la pantalla que esté actualmente en la memoria de video.

Ejemplo :
escribe_mensaje(descripción,5,5);
Esta instrucción coloca el contenido de la variable "descripción" en el renglón #5, columna #5 de la pantalla que actualmente ocupe la memoria de video.

Procedimiento : scroll_mensaje
Parámetros : mensaje : palabra
var renglon : integer
columna : integer
margen_derecho : integer

despliega : boolean

Descripción:
Esta rutina se encarga de colocar mensajes en la pantalla que actualmente se encuentre en la memoria de video, en el renglón que especifique a través del parámetro "renglón", de forma tal que el scrolling de mensajes se hace automáticamente por responsabilidad de esta rutina. El parámetro "margen_derecho" indica que el sistema debe limpiar con espacios blancos al renglón actual hasta la posición indicada por dicho parámetro. El parámetro "despliega" indica al sistema si se hace o no el despliegue del mensaje en el monitor.

Ejemplo :
scroll_mensaje(dato,renglon_pantalla,34,79,true);
Esta instrucción coloca el contenido de la variable "dato" en el número de renglón especificado por la variable "renglón_pantalla", cuyo valor se incrementará en 1 después de ejecutar esta instrucción. El mensaje se desplegará a partir de la columna 34 hasta la columna 79, añadiendo espacios en blanco hasta dicha posición en caso de que sea necesario. El parámetro "true" indica que si debe hacerse el despliegue del mensaje.

Procedimiento : desactiva_mensaje
Descripción:
Esta rutina se encarga de colocar el color original en el último mensaje que se haya activado en la pantalla que actualmente se encuentra en la memoria de video.

Procedimiento : activa_mensaje
Parámetros : mensaje : cadena
renglón,
columna : integer

Descripción:
Esta rutina se usa para resaltar el color de mensajes en la pantalla que se encuentre en la memoria de video. El parámetro "mensaje" indica el mensaje a activar, y su posición en la pantalla se indica a través de los parámetros "renglón" y "columna".

Ejemplo :
activa_mensaje('Mensaje del sistema',5,6);
Esta instrucción se encarga de colocar en negro al mensaje "Mensaje del sistema" que se ubica en el renglón 5, columna 6, de la pantalla que se encuentre actualmente en la memoria de video.

Nombre de la unidad : d_vta_1
Unidades requeridas :
crt,
man_vta,
d_vta_2,
d_vta_3

Descripción general :
El propósito de las rutinas de esta unidad consiste en la generación inicial de las ventanas del sistema. Estas ventanas, al ser generadas por primera vez, quedan presentes en memoria dinámica.

Rutinas de esta unidad:

Procedimiento : inicio_1
Descripción:
Esta rutina se encarga de crear en memoria dinámica a la pantalla principal del sistema.

Procedimiento : ini_1_ventanas
Descripción:
Esta rutina se encarga de crear al inicio de la ejecución del sistema, a aquellas ventanas fundamentales para el sistema. De esta forma, estas ventanas siempre permanecen presentes en memoria.

Procedimiento : gen_vta
Parámetros : número_ventana : integer
Descripción:
Esta rutina se encarga de crear en memoria dinámica a las ventanas del sistema. Cada ventana se identifica por un número, por lo que al tratar de desplegar una ventana, el sistema revisa si esta ventana está presente en memoria dinámica. En caso negativo, esta rutina llama a la rutina correspondiente que define a dicha ventana para construirla y dejarla presente en el sistema. Hecho lo anterior, dicha ventana permanecerá presente hasta el fin de ejecución del sistema.

Esta rutina es utilizada por la rutina "construye_ventana".

Ejemplo : gen_vta(3);
Esta instrucción se encarga de generar a la ventana #3.

Nombre de la unidad : man_1_vta
Unidades requeridas :
crt,
man_vta,
d_vta_1

Rutinas de esta unidad:

Procedimiento : construye_ventana
Parámetros : número_ventana : integer

Descripción:
Esta rutina se encarga de ejecutar las siguientes acciones:
- Crear en memoria dinámica a la ventana.
- Colocar a la ventana en el tope de la pila de ventanas del sistema.
- Salvar la región de video que quede debajo de la ventana.
- Desplegar la ventana en el monitor.

Ejemplo : construye_ventana(4);

Nombre de la unidad : tiempos

Descripción general :

Las rutinas de esta unidad tienen como propósito manejar la hora y fecha de la computadora, y desplegarlos en el monitor.

Rutinas de esta unidad:

Procedimiento : tim_inicial

Descripción:

Esta rutina se encarga el tiempo de la microcomputadora al inicio de ejecución del sistema.

Procedimiento : reloj

Parámetros : columna,
renglón : integer

Descripción:

Esta rutina se encarga de desplegar la hora, minutos y segundos del sistema. La información aparecerá en la columna y renglón especificados. La información se actualizará constantemente.

Ejemplo : reloj(33,12);
Esta instrucción mostrará la hora del sistema,en la columna 33, renglón 12 del monitor.

Procedimiento : tiempo

Descripción:

Esta rutina se usa para calcular el tiempo total transcurrido durante la ejecución del sistema.

Nombre de la unidad : arboles

Descripción general :

Las rutinas de esta unidad tienen como propósito manejar las estructuras de datos dinámicas del sistema. Estas estructuras de datos son: árboles y listas doblemente encadenadas.

Rutinas de esta unidad:

Procedimiento : ins_arbol

Parámetros : var nombre_árbol : ap_nodo
llave_nodo : palabra
información_nodo : palabra
estado_nodo : char
var indica_cambio: boolean

Descripción:

Esta rutina se encarga de insertar un nodo en el árbol especificado através del parámetro "nombre_árbol". La llave del nodo se especifica a través del parámetro "llave_nodo". Si se añade un nodo que ya existe en el arbol, entonces se cambiará el contenido de "información_nodo". Si se llega a presentar la situación actual, el parámetro "indica_cambio" regresará el valor "verdadero", de lo contrario regresará el valor "falso" indicando que el nodo no existia de antemano y se dió de alta por primera vez. En estado_nodo se indican los siguientes valores dependiendo del tipo de árbol:

'f': modificador forzoso en palabras clase.
'o': modificador opcional en palabras clase.
' ': nodo normal en palabras entidad, diccionarios de datos y ambientes de programación.
'b': nodo eliminado.

Ejemplo :

ins_arbol(arb_clase,'número','num','f',cambio);

Esta instrucción se encarga de almacenar un nodo en el árbol cuyo nombre es "arb_clase". La llave del nodo es "número" y la información a almacenar es su abreviación: 'num'. La 'f' indica que se trata de una palabra clase de modificador forzoso. Si este nodo no existia previamente en el árbol, el parámetro "cambio" regresará el valor "falso". Si el nodo ya existía, cambiará sus valores por los que se están indicando en esta

instrucción, y el parámetro "cambio" regresará el valor "verdadero".

Función : **consulta_arbol** : **palabra**
Parámetros : var nombre_árbol : ap_nodo
llave_nodo : palabra

Descripción:
Esta función da como resultado la información almacenada en el nodo cuya llave se especifica en el parámetro "llave_nodo" y que pertenece al árbol especificado en el parámetro "nombre_árbol". Si no existe algún nodo con dicha llave, entonces esta rutina dará como resultado a: ''

Ejemplo :
abreviación:=consulta_arbol(arb_clase,'número');
En este ejemplo, se busca la abreviación de la palabra 'número' en el árbol "arb_clase".

Función : **busca_arbol** : **boolean**
Parámetros : var nombre_árbol : ap_nodo
llave_nodo : palabra

Descripción:
Esta rutina se encarga de indicar si un nodo, con la llave especificada, exite en el árbol indicado. En caso afirmativo, contesta con "verdadero"; en caso negativo, contesta con "falso". Si el nodo fue borrado con anterioridad, contestará con "falso".

Ejemplo :
if busca_arbol(arb_clase,'cliente') then ...

Función : **busca_modificador**: **char**
Parámetros : var nombre_árbol : ap_nodo
llave_nodo : palabra

Descripción:
Esta rutina se encarga de consultar el campo de estado del nodo cuya llave se especifica a través del parámetro "llave_nodo". Si el nodo no existe en el árbol especificado en el parámetro "nombre_árbol", la rutina contesta con el valor ''.

Ejemplo :
writeln(busca_modificador(arb_clase,'número');

Procedimiento : pon_en_lista
Parámetros : var nombre_árbol : ap_nodo
var cab_l_i_s_t_a_p_u_n_t_a_d_o_r: ap_lista

Descripcion:
Esta rutina se encarga de tomar a un elemento de una lista doblemente encadenada y lo asocia con un nodo de un árbol, cuyo nombre se especifica en el parámetro "nombre_árbol".
Esta rutina es utilizada por la rutina "arma_lista".

Procedimiento : arma_lista
Parámetros : var nombre_árbol : ap_nodo
var nombre_lista : ap_lista

Descripción:
Esta rutina se encarga de construir una lista doblemente encadenada, y cada elemento apuntará a un nodo de un árbol en especial. El parámetro "nombre_lista" indica el nombre de la lista doblemente encadenada, y el parámetro "nombre_árbol" indica el nombre del árbol al que se asociará dicha lista.

Ejemplo :
arma_lista(arb_clase,lis_clase);
Esta instrucción asocia a la lista doblemente encadendada "lis_clase" con los nodos del árbol "lis_clase".

Procedimiento : recorre_lista
Parámetros : var nombre_lista : ap_lista

Descripción:
Esta rutina se encarga de navegar a la lista cuyo nombre se indica a través del parámetro "nombre_lista".

Ejemplo :
recorre_lista(lis_entidad);

Procedimiento : inicia_cabeza
Parámetros : var nombre_lista : ap_lista

Descripción:
Esta rutina se encarga de colocar al apuntador de la cabeza de la lista en el primer elemento de ésta.

Procedimiento : ven_listas
Parámetros : número_ventana : integer
var nombre_lista : ap_lista

Descripción:
Esta rutina se encarga de manejar una ventana que permita explorar a los elementos de una lista, de forma tal que la exploración se haga en segmentos, recorriendo hacia arriba o hacia abajo a dicha lista.

Ejemplo :
```
ven_listas(2,lis_entidad);
```

En este ejemplo, se usa a la ventana 2 para desplegar a los elementos de la lista de entidades. Recuérdese que cada lista está asociada a un árbol, por lo que en realidad se está teniendo acceso a los nodos del árbol a través de la lista.

Función : semejanza : boolean
Parámetros : cad_bus,cad_fte : palabra

Descripción:
Esta función se encarga de determinar si existe alguna semejansa entre 2 palabras, las cuales se especifican a través de los parámetros "cad_bus" y "cad_fte". Si existe alguna semejanza, la función genera el resultado "verdadero" ; de lo contrario, genera el resultado "falso".

Ejemplo :
```
if semejanza('número','numeración') then
     writeln('palabras semejantes') else
     writeln('no se parecen');
```

A través de este ejemplo se está verificando la semejanza entre las palabras 'número' y 'numeración'.

Procedimiento : crea_arb_abrev
Parámetros : var nombre_árbol_fuente : ap_nodo
var nombre_árbol_destino: ap_nodo

Descripción:
Esta rutina se encarga de revisar los nodos de los árboles , tomar sus campos de información y generar o actualizar un árbol en el que dichos campos serán llaves.

Lo anterior se usa para generar el árbol de abreviaciones. Para ello, se revisan tanto al árbol de palabras clase , como el de palabras entidad. En estos, las abreviaciones están almacenadas en el

campo de información. En el arbol de abreviaciones, (el que se crea y actualiza con esta rutina), dichas abreviaciones vienen a ser ahora las llaves, mientras que las palabras clase y entidad pasan a ser los campos de informacion.

Ejemplo :
crea_arb_abrev(arb_clase,arb_abrevia);

Procedimiento : cuenta_arbol
Parametros : var nombre_árbol : ap_nodo
var cantidad_nodos : integer
Descripcion:
Esta rutina se encarga de contar los nodos de un árbol, cuya identificación se indica por medio del parámetro "nombre_arbol". La cantidad de nodos se almacena en el parámetro "cantidad_nodos".

Ejemplo :
cuenta_arbol(arb_clase,total_nodos);
Esta instrucción cuenta los nodos del árbol "arb_clase" y la cantidad se almacena en la variable "total_nodos".

Función : abrev_repetida : boolean
Parámetros : llave,
abreviación : palabra

Descripción:
Esta rutina se encarga de revisar si la abreviación que se quiere dar de alta ya existe asociada a otra palabra en el acervo del sistema.
Si la abreviación ya existe, la rutina contesta "verdadero", de lo contrario contesta "falso".

Ejemplo :
if abrev_repetida('cliente','cte') then ...

Procedimiento : borra_arbol
Parámetros : var nombre_árbol : ap_nodo

Descripción:
Esta rutina se encarga de borrar fisicamente a todos los nodos del árbol cuyo nombre se especifica en el parámetro "nombre_árbol". La memoria dinámica ocupada por el árbol se libera en el sistema para ser usada por otros nodos o elementos dinámicos.

Ejemplo :
borra_arbol(arb_abrev);

Procedimiento : borra_lista
Parámetros : var nombre_lista : ap_lista

Descripción:
Esta rutina se encarga de borrar fisicamente a todos los elementos de la lista doblemente encadenada cuyo nombre se especifica en el parámetro nombre_lista. La memoria dinámica ocupada por la lista se libera para ser utilizada por otros árboles o listas.

Ejemplo :
borra_lista(lis_clase);

Procedimiento : arma_arb_abrev
Parámetros : var nombre_árbol : ap_nodo

Descripción:
Esta rutina se encarga de actualizar al árbol de abreviaciones. Para ello, toma al árbol cuyo nombre se especifica en el parámetro "nombre_árbol", lo recorre y de cada nodo toma al campo de información y lo da de alta como llave en el árbol de abreviaciones. La llave del árbol especificado a través del parámetro "nombre_árbol" se almacena en el campo de información del árbol de abreviaciones.

Ejemplo :
arma_arb_abrev(arb_clase);

Nombre de la unidad : pilas

Unidades requeridas : crt

Descripción general :

Las rutinas de esta unidad tienen como propósito manejar las pilas del sistema que son usadas para controlar la operación de las ventanas del sistema, así como para llevar el control de la secuencia de las palabras que forman parte de la definición del dato.

Rutinas de esta unidad:

Procedimiento : push_stack
Parámetros : estado_autómata : integer
palabra_esperada: palabra

Descripción:

Esta rutina se usa para almacenar en una pila a las palabras que forman parte de la descripción del dato a nombrar. En dicha pila, se almacena también el estado del automata de sintaxis en el que se está escribiendo la palabra a guardar.

El parámetro "estado_autómata" representa al estado en el que se escribe la palabra esperada por el autómata. El parámetro "palabra_esperada" representa a la palabra que se escribió en ese estado del autómata.

Ejemplo :

```
push_stack(0,'él');
```

Procedimiento : enciende_mensaje
Descripción:

Esta rutina se encarga de tomar a la palabra que se encuentra en el tope de la pila de palabras de la descripción del dato, y le cambia el color.

Esta rutina se utiliza en la rutina de correción de texto. Su propósito consiste en permitir al usuario ver las palabras que se eliminarán al momento de hacer la correción del texto de descripción del dato a nombrar.

Procedimiento : apaga_mensaje
Descripción:

Esta rutina se encarga de colocar el color original de la palabra que se encuentra en el tope de la pila de las palabras que forman parte de la descripción del dato a nombrar.

Nombre de la unidad : scanner

Unidades requeridas :
crt,
dos

Descripción general :
El propósito de esta unidad consiste en proveer rutinas que se encargan del manejo del teclado de la microcomputadora.

Rutinas de esta unidad:

Procedimiento : sin_cursor
Descripción:
Esta rutina se encarga de quitar al cursor del monitor de la computadora.

Procedimiento : con_cursor
Descripción:
Esta rutina se encarga de desplegar al cursor en el monitor de la computadora.

Función : oprime_esc_int : integer
Descripción:
Esta rutina se encarga de revisar que la tecla que se oprima sea "?" o "ESC". Si no es ninguna de éstas, ignora la tecla oprimida y espera a que se oprima la tecla correcta.
Como resultado, esta rutina genera el siguiente código para las teclas anteriores:

0 para ?
1 para ESC

Función : letra_a_m : integer
Descripción:
Esta rutina se encarga de revisar que la tecla que se oprima sea "a" o "m". Si no es ninguna de éstas, ignora la tecla oprimida y espera a que se oprima la tecla correcta.
Como resultado, esta rutina genera el siguiente código para las teclas anteriores:

0 para "a"
1 para "m"

Función : espera_o_f : char
Descripción:
Esta rutina se encarga de revisar que la tecla que se oprima sea "o" o "f". Si no es ninguna de éstas, ignora la tecla oprimida y espera a que se oprima la tecla correcta.

Función : espera_b_o_f : char
Descripción:
Esta rutina se encarga de revisar que la tecla que se oprima sea "o", "f" o "ESC". Si no es ninguna de éstas, ignora la tecla oprimida y espera a que se oprima la tecla correcta

Procedimiento : cualquier_tecla
Descripción:
Esta rutina genera una espera, la cual termina al momento de oprimir cualquier tecla.

Procedimiento : scan_editor
Parámetros : var mensaje_leido : string
var estado_scanner : integer
longitud_máxima_mensaje : integer
inhibe_caracter_numérico: boolean
indica_eco : boolean
Descripción:
Esta rutina se encarga de hacer la función de un scanner del teclado. A través de ella, el sistema revisa las teclas que se oprimen, generando un mensaje que puede ser editado al borrar los caracteres oprimidos a través de la tecla "back_space". El mensaje generado se almacenará en el parámetro "mensaje_leido". La barra espaciadora se considera con la misma funcion que la tecla "enter" ó "return", con la que se termina de hacer la revisión del teclado.

A través del parámetro estado_scanner se indica si la longitud del mensaje es diferente de 0. Si el valor de este parámetro es de 0, indica que se oprimió la tecla "enter" o "barra_espaciadora" o "return". Si se oprimió la tecla "ESC", entonces este parámetro tendrá valor 1, y la longitud del mensaje será 0.

A través del parámetro "longitud_máxima_mensaje" se especifica al scanner el número máximo de caracteres que compondrán al mensaje, una vez excedido el máximo, los demás caracteres no serán tomados en cuenta.

A través del parámetro "inhibe_caracter_numérico", el scanner tomará en cuenta a los caracteres numéricos (en caso de tener valor "verdadero") o no los considerará en el mensaje (en caso de tener valor "falso").

El parámetro "indica_eco" sirve para indicar al scanner si se debe hacer el eco de cada tecla oprimida en el monitor: Si vale "verdadero", se

hará el eco; si vale "falso", no se hace el eco.

Ejemplo :

```
scan_editor(descripcion,edo,40,true,false);
```

Con esta instrucción, se indica que el mensaje que se teclee será almacenado en la variable "descripción". El estado final del scanner se almacenará en la variable "edo". La longitud máxima del mensaje a analizar será de 40 caracteres. El siguiente parámetro indica que los caracteres numéricos no se tomarán en cuenta. El último parámetro indica que no se hará el eco en el monitor al momento de oprimir a cada tecla.

Procedimiento : scan_numro

Parámetros :

```
var mensaje_leido          : string
var estado_scanner         : integer
longitud_máxima_mensaje : integer
```

Descripción:

Esta rutina se encarga de analizar exclusivamente a las teclas que representan a los caracteres numéricos. Los caracteres alfabéticos se omiten en el análisis. El mensaje leído se almacenará en el parámetro "mensaje_leido".

A través del parámetro "estado_scanner" el sistema indicará si el análisis terminó de hacerse debido a que se oprimió la tecla "barra_espaciadora", "enter" o "return" o si el mensaje tiene longitud 0 por haber oprimido a la tecla "ESC". Si el sistema contesta a través de este parámetro con el valor 0, significa que se oprimió la tecla "enter", "return" o "barra_espaciadora"; si contesta con el valor 1, significa que se oprimió a la tecla "ESC" y que el mensaje tiene longitud de 0 caracteres.

Mediante el parámetro "longitud_máxima_mensaje" se indica al scanner el número máximo de caracteres a considera en el mensaje; si el número es excedido, los demás caracteres numéricos no se añaden al mensaje.

Ejemplo :

```
scan_numro(numero,edo,2);
```

Con esta instrucción, se hace el análisis del teclado, de forma tal que el mensaje detectado se almacenará en la variable "número". El estado final del scanner se almacenará en la variable "edo". El número de caracteres a analizar será de 2 como máximo.

Procedimiento : scan_caresp

Parámetros : var mensaje_leido : string
var estado_scanner : integer
longitud_máxima_mensaje : integer
inhibe_caracter_numerico: boolean

Descripción:

Esta rutina se encarga de hacer la función de un scanner del teclado, considerando en el análisis sólo a los caracteres especiales (¡¿#$%/&**()_-+=Çç'?!":) y numericos.

A través de ella, el sistema revisa las teclas que se oprimen, generando un mensaje que puede ser editado al borrar los caracteres oprimidos a través de la tecla "back_space". El mensaje generado se almacenará en el parámetro "mensaje_leído". La barra espaciadora se considera con la misma función que la tecla "enter" o "return", con la que se termina de hacer la revision del teclado.

A través del parámetro estado_scanner se indica si la longitud del mensaje es diferente de 0. Si el valor de este parámetro es de 0, indica que se oprimió la tecla "enter" o "barra_espaciadora" o "return". Si se oprimió la tecla "ESC", entonces este parámetro tendrá valor 1, y la longitud del mensaje será 0.

A través del parámetro "longitud_máxima_mensaje" se especifica al scanner el número máximo de caracteres que compondrán al mensaje, una vez excedido el máximo, los demás caracteres no serán tomados en cuenta.

A través del parámetro "inhibe_caracter_numérico", el scanner tomará en cuenta a los caracteres numéricos (en caso de tener valor "verdadero") o no los considerará en el mensaje (en caso de tener valor "falso").

Ejemplo :

```
scan_caresp(car_especial,edo,1,true);
```

A través de esta instrucción, se hace la revisión del teclado, de forma tal que sólo se consideran a los caracteres especiales. El mensaje leído se almacenará en la variable "car_especial". El estado del scanner se almacenará en la variable "edo". El número de caracteres que formarán al mensaje será de "1". El último parámetro indica que sí se considerarán a los caracteres numéricos en el análisis del teclado.

Nombre de la unidad : man_arch
Unidades requeridas :
dos,
crt,
man_vta,
d_vta_1,
d_vta_2,
d_vta_3,
pilas

Descripción general:
Las rutinas de esta unidad tienen el propósito de manejar los archivos que forman parte del acervo del sistema de nomenclatura de datos.

Rutinas de esta unidad:

Función : **id_unidad_drive** : **palabra**
Parámetros : unidad_disco_actual : char

Descripción:
Esta función da como resultado la identificación del disco que se encuentra en la unidad especificada por el parámetro "unidad_disco_actual". La identificación estará formada por:
- Nombre del disco.
- Fecha de creación del nombre del disco.
- Hora de creación del nombre del disco.

Ejemplo :
```
identificacion_disco_duro:=id_unidad_drive('c');
```

Función : **nm_unidad_drive** : **palabra**
Parámetros : unidad_disco_actual : char

Descripción:
Esta función da como resultado el nombre del disco que se encuentra en la unidad de disco especificada por el parámetro "unidad_disco_actual".

Ejemplo :
```
writeln(nm_unidad_drive('a');
```

Procedimiento : **anade_palabra**
Parámetros : nombre_archivo : string
palabra_llave : palabra
información : palabra
estado_elemento : char

Descripción:
Esta rutina se utiliza para actualizar los archivos del sistema. Con ella, se dan de alta

nuevas palabras clase, entidad, así como diccionarios de datos y ambientes de programación.

El parámetro "nombre_archivo" se encarga de indicar el nombre del archivo a actualizar. Los demas parámetros constituyen la información a almacenar.

El parámetro "estado_elemento" indica el estado que tendrá dicho registro:

'b' indicara que dicho registro se considera como eliminado del sistema.

Ejemplo :
anade_palabra('entidad.dat','cliente','cte','b');

La instrucción anterior indica que la palabra entidad "cliente" se consiera como eliminada del acervo del sistema.

Procedimiento : trae_palabra
Parámetros : var palabra_llave : palabra
var información : palabra
var estado_elemento : char

Descripción:
Esta rutina se encarga de traer un registro del archivo que previamente fue abierto por el sistema. Cada registro que lee esta rutina se almacenerá en el árbol correspondiente.

En el parámetro "palabra_llave" esta rutina almacenará a la llave del registro. En el parámetro "información" almacenerá al campo de información del registro. Por último, en el parámetro "estado_elemento" almacenerá el estado del registro, el cual puede tener varios valores:

' ' : para palabras entidad, diccionarios de datos y ambientes de programación.
'f' : para palabras clase de modificador forzoso.
'o' : para palabras clase de modificador opcional.
'n' : para usuarios de sólo consulta.
'a' : para administradores de datos por área del negocio o proyecto.
'c' : para el administrador de datos corporativo.
'b' : indica que el registro debe considerarse como borrado o eliminado del acervo del sistema.

La lectura de los registros de cada archivo se hará en forma secuencial.

Ejemplo :
trae_palabra(pal_clase,abrev,edo);

Procedimiento : anade_regla
Descripción:
Esta rutina se encarga de actualizar el acervo de reglas de abreviación del sistema. A través de ella, se almacena un registro con los valores de la regla de abreviacion a actualizar. Dicho registro se almacenará al final del archivo de reglas de abreviación.

Procedimiento : trae_regla
Parámetros : nombre_ddds_ambiente : string
tipo_elemento : integer

Descripción:
Esta rutina se encarga de leer del archivo de reglas de abreviación el registro que contiene a las reglas de abreviación que se desee. Para ello, el parámetro "nombre_ddds_ambiente" especificará el nombre del diccionario de datos o ambiente de programación del que se quiere leer sus reglas de abreviación. El parámetro "tipo_elemento" indicará si se trata de un diccionario de datos o un ambiente de programación.

Ejemplo :
trae_regla('addict',1);

Nombre de la unidad : inicios

Unidades requeridas :
crt,
dos,
man_arch,
arboles

Descripción general :
Las rutinas de esta unidad tienen el propósito de inicializar al acervo del sistema.

Rutinas de esta unidad:

Procedimiento : ini_entidad
Descripción:
Esta rutina se encarga de generar el conjunto básico de palabras entidad del sistema. Esta rutina es usada por la rutina "ini_árboles".

Procedimiento : ini_ddds
Descripción:
Esta rutina se encarga de generar el conjunto básico de diccionarios de datos contemplados por el sistema. Esta rutina es usada por la rutina "ini_árboles".

Procedimiento : ini_amb_prog
Descripción:
Esta rutina se encarga de generar el conjunto básico de ambientes de programación contemplados por el sistema. Esta rutina es usada por la rutina "ini_arboles".

Procedimiento : ini_clase
Descripción:
Esta rutina se encarga de generar el conjunto básico de palabras clase del sistema. Esta rutina es usada por la rutina "ini_arboles".

Procedimiento : ini_arboles
Descripción:
Esta rutina se encarga de las funciones siguientes:

- Inicializar al árbol de palabras clase con las palabras clase almacenadas en disco. Si el archivo correspondiente no existe, genera al conjunto básico de palabras clase mediante la rutina "ini_clase".

- Inicializar al árbol de palabras entidad con las palabras entidad almacenadas en disco.Si El archivo correspondiente no existe, genera

al conjunto básico de palabras entidad mediante la rutina "ini_entidad".

- Inicializar al árbol de nombres de diccionarios de datos almacenados en disco. Si el archivo correspondiente no existe, genera al conjunto básico de nombres de diccionarios de datos mediante la rutina "ini_ddds".

- Inicializar al árbol de nombres de ambientes de programación almacenados en disco. Si el archivo correspondiente no existe, genera al conjunto básico de nombres de ambientes de programación mediante la rutina "ini_amb_prog".

Nombre de la unidad : seguridad

Unidades requeridas :

crt,
dos,
scanner,
man_vta,
man_l_vta,
d_vta_1,
d_vta_2,
d_vta_3,
arboles,
man_arch

Descripción general :

Las rutinas de esta unidad tienen la función de seguridad del sistema. Ellas se encargan de la autificación de usuarios, así como permitir sólo las acciones a las cuales se les ha autorizado al momento de darlos de alta en el sistema.

Rutinas de esta unidad:

Función : autentificacion:boolean
Descripción:
Esta función se encarga de controlar el acceso al sistema. Se encarga de revisar el nombre del usuario y la clave de acceso secreta.

Ejemplo :
```
if autentificación then writeln('acceso autorizado')
```

Función : autorizado : boolean
Parámetros : nivel_autorización : palabra
Descripción:
Cada vez que se intente actualizar al acervo del sistema, esta rutina revisa que el usuario cuenta con el nivel de privilegios que le autoriza a hacer la operación de actualización.

Función : verificado:boolean
Descripcion:
Esta función se encarga de preguntar al usuario si está seguro de llevar a cabo la actualización. Como resultado, esta función da "verdadero" en caso de que el usuario haya contestado que sí desea realizar la actualización; el valor "falso" se da como resultado en caso de que el usuario no haya querido realizar la actualización.

Ejemplo :
if verificado then anade_regla;

Función : mismo_disco:boolean

Descripción:

Esta función se encarga de revisar si la identificación del disco duro en el que se encuentra instalado el sistema es igual a la identificación que se usó durante el proceso de instalación del sistema.

Como resultado, indica "verdadero" en caso de que coincidan las identificaciones; en caso contrario, el resultado de esta función es "falso".

Ejemplo :
if mismo_disco then ...

Nombre de la unidad : reglas

Unidades requeridas :

crt,
man_vta,
man_1_vta,
man_arch,
d_vta_1,
d_vta_2,
scanner,
arboles,
seguridad

Descripción general:

Las rutinas de esta unidad tienen el propósito de definir, consultar, eliminar y actualizar a las reglas de abreviación.

Rutinas de esta unidad:

Procedimiento : display_regla

Descripcion:

Esta rutina se encarga de mostrar en el monitor el estado de la regla de abreviación actual.

Ejemplo :

```
if cambia_regla then display_regla;
```

Procedimiento : autom_reglas

Descripcion:

Esta rutina se encarga de llevar el control del autómata usado para la actualización de las reglas de abreviación pertenecientes a diccionarios de datos y ambientes de programación.

Procedimiento : inicia_reglas

Descripción:

Esta rutina se encarga de inicializar a la matriz de transición de estados que es usada por el autómata que se encarga de la actualización de las reglas de abreviación.

Procedimiento : recibe_numero

Parámetros :

```
var número_leído: integer
    longitud_máxima_número,
    valor_mínimo,
    valor_máximo     : integer
```

Descripción:

A través de esta rutina, el sistema se encarga de recibir del teclado un número, el cual se usa para actualizar reglas de abreviación tales como:

- Número máximo de caracteres por palabra
- Número mínimo de caracteres por palabra
- Número máximo de palabras por nombre

Asimismo, esta rutina se encarga de validar ciertas condiciones, como son:

- El número máximo de caracteres por palabra debe ser mayor al número mínimo.

- El número mínimo de caracteres por palabra debe ser menor al número máximo y a 2.

- El número máximo de palabras por nombre debe ser al menos mayor a 2.

Para estas validaciones se hace uso de los parámetros "valor_minimo" y "valor_maximo". El número ya validado se almacena en el parámetro "numero_leido". La cantidad de caracteres numéricos que formarán a dicho número se especifica en el parámetro "longitud_máxima_número".

Ejemplo : recibe_número(reglas.maximo,2,2,9);

Nombre de la unidad : man_imp

Unidades requeridas : dos, crt

Descripción general :

Las rutinas de esta unidad tienen como proposito formar una infraestructura que permita generar archivos reconocibles por GEM 1st WordPlus, y que son usadas por el sistema para generar los reportes del mismo.

Rutinas de esta unidad:

Procedimiento : muestra_buffer

Descripcion:

Esta rutina se encarga de mostrar en el monitor el contenido actual del buffer de generación de documentos.

Procedimiento : limpia_buffer

Descripcion:

Esta rutina se encarga de inicializar el contenido del buffer de generacion de documentos y el contador de caracteres del buffer.

Procedimiento : respalda_buffer

Descripción:

Esta rutina se encarga de respaldar el buffer de generación de documentos en el disco. La accion anterior se llevará a cabo en el momento en que se agote el espacio disponible en el buffer. Si es la primera vez que se respalda al buffer, esta rutina habre el archivo fisico asociado al documento y respalda al buffer.
Una vez respaldado el buffer, este es inicializado para tener nuevamente espacio disponible en él.

Procedimiento : scroll_texto

Parámetros : linea_texto : string
Número_renglones: integer

Descripción:

Esta rutina toma al parametro "linea_texto" (que representa un renglón en el documento que se desea generar) y le añade caracteres de control, de forma que el documento se almacene en forma de un archivo magnético reconocible por GEM 1st WORDPLUS. El parámetro "número_renglones" indica el número de cambios de renglón que se harán después de la linea de texto especificada en el primer parámetro.

Una vez añadidos los caracteres de control, esta rutina almacena dicha información en el buffer de

generación de documentos. Si se alcanza la capacidad máxima del buffer (2Kbytes) se manda ejecutar a la rutina "respalda_buffer" para que el buffer se respalde en disco y se inicialice para tener espacio libre y continuar generando al documento.

Procedimiento : imp_reloj
Descripción:
Esta rutina toma de la microcomputadora la fecha y hora actuales y los almacena en el buffer de generación de documentos.

Procedimiento : pon_header
Descripción:
Esta rutina se usa para iniciar la generación del documento. Manda limpiar el buffer de generación de documentos y posteriormente coloca los caracteres de control necesarios para que el documento sea reconocido por GEM 1st wordplus.

Procedimiento : pon_fin_pagina
Descripción:
Esta rutina coloca caracteres de control que indican el cambio de página. Si se alcanza la capacidad maxima del buffer, éste se respalda y posteriormente se continúan almacenando los caracteres restantes en el buffer ya vacio.

Procedimiento : guarda_buffer
Descripción:
Esta rutina coloca los caracteres de control necesarios para indicar el fin de documento y almacena el buffer de generación de documentos en disco. Si el buffer no había sido respaldado con anterioridad (mediante la rutina respalda_buffer), entonces se encarga de asociar el nombre lógico al archivo fisico y lo abre para posteriormente almacenar el buffer. Hecho lo anterior, cierra definitivamente el archivo.

Procedimiento : formatea
Parámetros :
men_formateo:string
pos:integer
men_entrada:string
Descripción:
Esta rutina se encarga de tomar al parámetro "texto" y ajustarlo para que inicie en la columna especificada en el parámetro "número de columna". Esta rutina se usa para lograr formar columnas de presentación de datos.

Nombre de la unidad : man_rep

Unidades requeridas :
crt,
man_imp,
arboles,
man_arch

Descripción general :
Las rutinas de esta unidad tienen el propósito de generar los reportes del sistema, en forma de archivos que sean reconocibles por GEM 1st WordPlus.

Rutinas de esta unidad:

Procedimiento : imprime_arbol
Parámetros :
nombre_árbol : árbol
Descripción:
Esta rutina se encarga de navegar al árbol especificado y copia en el buffer de generación de documentos el contenido de cada nodo del árbol.

Procedimiento : imprime_borrados
Parámetros :
nombre_árbol : árbol
Descripción:
Esta rutina se encarga de navegar el árbol especificado y copia en el buffer de generación de documentos el contenido de los nodos del árbol que se encuentren marcados como eliminados.

Procedimiento : cuenta_borrados
Parámetros :
nombre_árbol : árbol
cantidad_borrados : integer
Descripción:
Esta rutina cuenta el número de nodos eliminados en el árbol especificado. El resultado lo almacena en el parámetro "cantidad_borrados".

Procedimiento : imp_abrev
Descripción:
Esta rutina navega el árbol de abreviaciones especiales del sistema y copia el contenido de los nodos existentes en el buffer de generación de documentos. Asimismo, almacena en el buffer los encabezados pertinentes, como son:

scroll_texto('Lista de abreviaciones especiales',2)
scroll_texto('La lista se encuentra vacía',2)

Procedimiento : imp_clase
Descripción:
Esta rutina se encarga de navegar el árbol de palabras clase del sistema y copia el contenido de los nodos existentes en el buffer de generación de documentos. Asimismo, almacena los encabezados pertinentes, como son:

```
scroll_texto('Palabras clase y abreviaciones:',2)
scroll_texto('La lista se encuentra vacia',2)
```

Procedimiento : imp_entidad
Descripción:
Esta rutina se encarga de navegar el árbol de palabras entidad del sistema y copia el contenido de los nodos existentes en el buffer de generación de documentos. Asimismo, almacena los encabezados pertinentes, como por ejemplo:

```
scroll_texto('Palabras entidad y abreviaciones',2)
scroll_texto('La lista se encuentra vacia',2)
```

Procedimiento : imp_ddds
Descripción:
Esta rutina se encarga de navegar el árbol de nombres de diccionarios de datos y copia el contenido de los nodos existentes en el buffer de generación de documentos. Asimismo, genera los encabezados pertinentes.

Procedimiento : imp_amb_prog
Descripción:
Esta rutina se encarga de navegar el árbol de nombres de ambientes de programación y copia el contenido de los nodos existentes en el buffer de generación de documentos. Asimismo, genera los encabezados pertinentes, como son:

```
scroll_texto('Lista de ambientes de programacion,1)
```

Procedimiento : imp_reglas
Descripción:
Esta rutina se encarga de copiar en el buffer de generación de documentos las reglas de abreviación que se encuentren actualmente en el registro de reglas de abreviación del sistema.

Procedimiento : imp_eli_clase
Descripción:
Esta rutina copia el contenido de los nodos eliminados del árbol de palabras clase en el buffer de generación de documentos. También genera los encabezados pertinentes, tales como:

scroll_texto('Lista de palabras clase eliminadas')
scroll_texto('No se han eliminado elementos',1)

Procedimiento : imp_eli_entidad
Descripción:
Esta rutina copia el contenido de los nodos eliminados del árbol de palabras entidad en el buffer de generacion de documentos. Genera los encabezados pertinentes, tales como:

scroll_texto('Lista de palabras entidad eliminadas)
scroll_texto('No se han eliminado elementos,1)

Procedimiento : imp_eli_ddds
Descripción:
Esta rutina copia el contenido de los nodos eliminados del árbol de nombres de diccionarios de datos en el buffer de generacion de documentos. También genera los encabezados pertinentes, tales como:

scroll_texto('Lista de diccionarios de datos eliminados',1)
scroll_texto('No se han eliminado elementos')

Procedimiento : imp_eli_amb_prog
Descripción:
Esta rutina copia el contenido de los nodos eliminados del árbol de nombres de ambientes de programación en el buffer de generación de documentos. También se encarga de generar los encabezados pertinentes, tales como:

scroll_texto('Lista de ambientes de programación eliminados',1)
scroll_texto('No se han eliminado elementos')

Procedimiento : imp_nom_prim
Parámetros :
texto_nombre_primario,
texto_descripción_dato,
nombre_diccionario_datos : cadena
Descripción:
Esta rutina se encarga de copiar en el buffer de generación de documentos el contenido de los parámetros de la rutina. Estos parámetros representan lo siguiente:

"texto_nombre_primario" representa al nombre primario del dato.

"texto_descripción_dato" representa al texto de descripción del dato que se uso para generar al nombre primario.

"nombre_diccionario_datos" representa al nombre del diccionario de datos en el que se almacenarán el nombre primario y la descripción del dato. También representa al diccionario de datos cuyas reglas de abreviación se usaron para generar al nombre primario del dato.

Procedimiento : imp_nom_prog
Parámetros :
texto_nombre_programación,
texto_descripción_dato,
nombre_ambiente_programación : cadena

Descripción:
Esta rutina se encarga de copiar en el buffer de generación de documentos el contenido de los parámetros de la rutina. Estos parámetros representan lo siguiente:

"texto_nombre_programación" representa al nombre de programación del dato.

"texto_descripción_dato" representa al texto de descripción del dato que se usó para generar al nombre de programación.

"nombre_ambiente_programación" representa al nombre del ambiente de programación en el que se usará al dato, y cuyas reglas de abreviación se usaron para generar al nombre de programación.

Nombre de la unidad : auto_rep

Unidades requeridas :

crt,
scanner,
arboles,
man_arch,
man_vta,
d_vta_1,
d_vta_2,
d_vta_3,
man_1_vta,
man_imp,
man_rep

Descripción general :

Las rutinas de esta unidad se encargan de permitir al usuario definir la estructura de los reportes que generará a través del sistema.

Rutinas de esta unidad:

Función : sig_tecla : integer
Descripcion :

Esta función se encarga de sensar al teclado y sólo aceptar a las teclas:

flecha superior
flecha inferior
enter
esc

Esta función es usada por los autómatas de manejo de ventanas y teclado para la definición del formato de los documentos de reportes del sistema de nomenclatura de datos.

Procedimiento : ini_aux_rep
Descripción:

Esta rutina se encarga de inicializar a la matriz de transición de estados del autómata que se utiliza para la definición del formato de documentos de reportes, indicando las reglas de abreviación de diccionarios de datos y ambientes de programación que aparecerán en dichos documentos.

Procedimiento : guarda_reglas_rep
Parámetros :

nombre_regla:string

Descripción:

Esta rutina se encarga de almacenar el nombre del diccionario de datos o ambiente de

programacion, cuyas reglas de abreviación se desea que aparezcan en el documento de reportes definido por el usuario.
Si el nombre especificado a través del parametro "nombre_regla" no está en la lista, se da de alta. Si el nombre ya existía con anterioridad, entonces se desecha de la lista.
Lo anterior tiene la finalidad de permitir al usuario total libertad de incluir o desechar nombres de la lista de reglas de abreviación a considerar en el documento de reportes.
Esta rutina, una vez hecho lo anterior, llama a las rutinas apropiadas para desplegar en el monitor a la lista de reglas de abreviación a considerar en el documento de reportes.

Procedimiento : aut_aux_rep
Descripción:
Esta rutina representa al autómata encargado de manejar los menús y teclado para la selección de los nombres de diccionarios y ambientes de programación que se desea que sus reglas de abreviación aparezcan en el documento de reportes.

Procedimiento : ini_rep
Descripción:
Esta rutina inicializa a la matriz de transición de estados del autómata que controla los menús y teclado para la definición del formato del documento de reportes.

Procedimiento : ini_orden_imp
Descripción:
Esta rutina se encarga de inicializar a la lista de reportes que usa el sistema para la generación del documento de reportes. La inicialización da como resultado que la lista tenga como primer elemento al reporte de nombres generados durante la sesión. Los siguientes elementos se encuentran vacíos, de forma tal que mediante el autómata de definición de reportes, el usuario es capaz de añadir elementos a la lista.
Esta rutina también se encarga de generar una lista auxiliar para llevar los nombres de los reportes a generar:

```
[0]:='                    '
[1]:='Palabras clase      '
[2]:='Palabras  entidad   '
[3]:='Diccionarios de datos'
[4]:='Ambientes de programación'
[5]:='Reglas de abreviación'
[6]:='Abreviaciones especiales'
```

[7]:='Eliminaciones
[8]:='Nombres generados

Procedimiento : ini_tipo_imp
Descripción:
Esta rutina se encarga de generar una lista auxiliar para llevar los nombres de los reportes a generar:

```
[0]:='                                  '
[1]:='Palabras clase                    '
[2]:='Palabras entidad                  '
[3]:='Diccionarios de datos             '
[4]:='Ambientes de programación         '
[5]:='Reglas de abreviación             '
[6]:='Abreviaciones especiales          '
[7]:='Eliminaciones                     '
[8]:='Nombres generados                 '
```

Procedimiento : guarda_orden_rep
Parámetros :
reporte_tipo : integer
Descripción:
Esta rutina es usada por el sistema para llevar en una lista ordenada, los reportes tipo que formarán parte del documento de reportes. En esta lista, se almacena la clave de cada reporte tipo. Si el valor del parámetro "reporte_tipo" ya existe con anterioridad en la lista, entonces esta rutina se encarga de eliminarla de la lista. Si por el contrario, no existe con anterioridad, entonces se incluye al final de la lista.

Procedimiento : autom_reps
Descripción:
Esta rutina representa al autómata que se encarga de llevar el control de los menús y teclado para la definición del documento de reportes que el sistema genera. En este autómata, se pide al usuario que especifique el nombre del archivo que se almacenará en disco y que contendrá al documento de reportes. Mediante este autómata el usuario puede escoger uno o varios reportes tipo, de acuerdo a la lista siguiente:

```
[0]:='                                  '
[1]:='Palabras clase                    '
[2]:='Palabras entidad                  '
[3]:='Diccionarios de datos             '
[4]:='Ambientes de programación         '
[5]:='Reglas de abreviación             '
[6]:='Abreviaciones especiales          '
[7]:='Eliminaciones                     '
[8]:='Nombres generados
```

Si el usuario desea incluir al reporte de reglas de abreviación, este autómata llama al autómata representado por la rutina "autom_aux_rep".

Nombre de la unidad : abrcion

Unidades requeridas :

```
crt,
scanner,
pilas,
man_vta,
man_1_vta,
reglas,
arboles,
man_arch,
tiempos,
man_imp,
man_rep,
auto_rep
```

Descripción general :

Las rutinas de esta unidad se encargan de generar las abreviaturas de las palabras claves de la definición del dato.

Rutinas de esta unidad:

Función : es_plural : boolean

Parámetros : palabra_fuente : palabra
var indica_español : boolean

Descripción:

Esta función se encarga de determinar si la "palabra_fuente" se encuentra en plural, y si de acuerdo a las reglas de construcción del plural, se trata de una palabra del idioma español o no. En caso de que si sea una palabra del idioma español, el parámetro "indica_español" tendrá valor "verdadero", de lo contrario tendrá valor "falso".

Ejemplo :

```
if es_plural('clientes',español) then
   writeln('palabra en plural');
```

Función : quita_plural : palabra

Parámetros : palabra_fuente : palabra
despliega_resultado: boolean

Descripción:

Función toma al parámetro "palabra_fuente" y como resultado genera el singular de la palabra. El parámetro "despliega_resultado" hace que se despliegue la palabra en singular (en caso de "verdadero") o no (en caso de "falso").

Ejemplo :

```
pal_singular:=quita_plural('clientes',false);
```

En este ejemplo, la función tomará a la palabra "clientes" y su singular (cliente) se asignará a la variable "pal_singular". No se despliega el resultado de esta acción en el monitor.

Procedimiento : prep_abrevia
Descripción:
Esta rutina toma a las palabras de la descripción del dato a nombrar, elimina artículos, preposiciones y contracciones, y coloca a las palabras claves en el orden que se haya indicado en las reglas de abreviación existentes en ese momento.

Procedimiento : autom_abrevia
Descripción:
Esta rutina se encarga de controlar al autómata de generación de abreviaciones. Entre las funciones que realiza este autómata, están:

-Selección de diccionario de datos o ambiente de programación, cuyas reglas habrán de usarse para generar los nombres primarios y de programación.

-Colocación de palabras claves en el orden especificado en las reglas de abreviación.

-Control de las rutinas encargadas de aplicar las reglas de abreviación.

-Control de las ventanas del menú de generación de abreviaciones y nombres.

Procedimiento : ini_a_abrevia
Descripción:
Esta rutina se encarga de inicializar a la matriz de transición de estados que es usada por el autómata de generación de abreviaciones.

Nombre de la unidad : menus

Unidades requeridas :

crt,
dos,
scanner,
man_vta,
man_1_vta,
d_vta_1,
d_vta_2,
d_vta_3,
arboles,
pilas,
reglas,
seguridad,
man_arch,
tiempos,
man_imp,
man_rep,
auto_rep,
abrcion

Descripción general:

Las rutinas de esta unidad se encargan del manejo del menú principal del sistema de nomenclatura de datos.

Rutinas de esta unidad:

Procedimiento : autom_menus

Descripción:

Esta rutina se encarga de llevar el control del autómata de los menús del sistema. Entre las funciones contempladas en este autómata están:

- Ayudas en línea para el usuario.
- Altas, bajas y cambios en:
  - palabras clase.
  - palabras entidad.
  - reglas de abreviacion.
  - diccionarios de datos.
  - ambientes de programación.
- Corrección del texto de la descripción.
- Salir del sistema o del menú.
- Información del sistema.

Procedimiento : ini_menus

Descripción:

Esta rutina se encarga de inicializar a la matriz de transición de estados del autómata de los menús del sistema.

Procedimiento : corrige_texto

Descripción:

Esta rutina permite al usuario eliminar palabras de la descripción del dato a nombar. Esta rutina hace todas las actualizaciones pertinentes en las estructuras de datos asociadas a la descripción del dato en forma automática.

En esta rutina se lleva el control del teclado, de forma tal que las flechas del teclado indican cuáles palabras habrán de eliminarse. La tecla "enter" se usa para aceptar la eliminación de las palabras. La tecla "ESC" se usa para abortar la eliminación de las palabras de la descripción.

Nombre de la unidad : morfemas

Unidades requeridas :

arboles,
inicios,
man_arch,
abrcion,
seguridad,
man_vta,
man_1_vta,
d_vta_1,
d_vta_2,
d_vta_3

Descripción general :

Las rutinas de esta unidad tienen el propósito de realizar el análisis morfológico de sustantivos y adjetivos. De esta forma, se evita tener un gran diccionario. Los gramemas característicos de sustantivos y adjetivos se manejan a través del autómata que se describe en la sección 5.4.9. (Descripción del autómata encargado de realizar el análisis morfológico de las palabras que forman parte de la definición del dato)

Rutinas de esta unidad:

Procedimiento : ini_morf

Descripción:

Esta rutina se encarga de inicializar la matriz de transición de estados del autómata descrito en la sección 5.4.9, el cual se encarga del análisis morfológico.

Función : autom_morf : integer

Parámetros : tipo_análisis = (sustantivo,adjetivo)
palabra_a_analizar : palabra

Descripción:

Esta función representa al autómata encargado de efectuar el análisis morfológico. Mediante el parámetro "palabra_a_analizar" se especifica la palabra que habrá de analizar morfológicamente. Mediante el parámetro "tipo_análisis" se indica si se espera que la palabra sea un adjetivo o un sustantivo. Esta función genera 3 posibles resultados:

0 --> "No corresponde al tipo".
1 --> "Corresponde al tipo".
2 --> "No se puede afirmar nada".

Función : **ver_sustantivo** : **boolean**
Parámetros : palabra_analizada : palabra

Descripción:

Esta función responde "verdadero" si la palabra analizada corresponde a un sustantivo. De lo contrario, responde "falso" y presenta una serie de diálogos para actualizar al diccionario de excepciones con la nueva palabra.

La función "ver_sustantivo" se encarga de explorar diferentes posibilidades para determinar si la palabra analizada corresponde realmente a un sustantivo. El autómata al que hace referencia, es el autómata descrito en la sección 5.4.9 (Descripción del autómata encargado de realizar el análisis morfológico de las palabras que forman parte de la definición del dato). Dicho autómata está construido mediante la función **autom_morf**.

Primero determina si la palabra se encuentra en plural, si esto es cierto, entonces la convierte al singular. Ya teniendo la palabra en singular, explora los siguientes casos para determinar si es un sustantivo:

1) Se afirma que es un sustantivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

   - La respuesta del autómata es "corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

2) Se niega que se trata de un sustantivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

3) Se niega que se trata de un sustantivo si se

cumplen las dos condiciones siguientes:

- La respuesta del automata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

- La respuesta del automata es "no corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

4) Se niega que se trata de un sustantivo si se cumplen las dos condiciones siguientes:

- La respuesta del autómata es "corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

- La respuesta de autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

Si las respuestas indican que no se trata de un sustantivo, como último recurso se busca a la palabra en el acervo de sustantivos que son excepciones a los gramemas característicos. Si no se le encuentra, entonces definitivamente no se trata de un sustantivo (puede actualizarse el acervo dando de alta a esta nueva palabra). Si se le encuentra, entonces se afirma que sí es un sustantivo.

Función : **ver_adjetivo : boolean**
Parámetros : palabra_analizada : palabra

Descripción:

Esta función responde "verdadero" si la palabra analizada corresponde a un adjetivo. De lo contrario, responde "falso" y presenta una serie de diálogos para actualizar al diccionario de excepciones con la nueva palabra.

La función "ver_adjetivo " se encarga de explorar diferentes posibilidades para determinar si la palabra analizada corresponde realmente a un adjetivo . El autómata al que hace referencia, es el autómata descrito en la sección 5.4.9 (Descripción del autómata encargado de realizar el análisis morfológico de las palabras que forman parte de la definición del dato). Dicho autómata está construido mediante la función **autom_morf.**

Primero determina si la palabra se encuentra en plural, si esto es cierto, entonces la convierte al singular. Ya teniendo a la palabra en singular, explora los siguientes casos para determinar si es un adjetivo:

1) Se afirma que es un adjetivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

   - La respuesta del autómata es "corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

2) Se niega que se trata de un adjetivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

3) Se niega que se trata de un adjetivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

   - La respuesta del autómata es "no corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "adjetivo".

4) Se niega que se trata de un adjetivo si se cumplen las dos condiciones siguientes:

   - La respuesta del autómata es "corresponde al tipo" al darle como parámetros la palabra a analizar y el tipo "sustantivo".

   - La respuesta de autómata es "no corresponde al tipo" o "no se puede afirmar nada" al darle como parámetros la palabra a analizar y el

tipo "adjetivo".

Si las respuestas indican que no se trata de un adjetivo, como ultimo recurso se busca a la palabra en el acervo de adjetivos que son excepciones a los gramemas caracteristicos. Si no se le encuentra, entonces definitivamente no se trata de un adjetivo (puede actualizarse el acervo dando de alta a esta nueva palabra). Si se le encuentra, entonces se afirma que si es un adjetivo.

Programa principal

Unidades requeridas:

crt,
dos,
scanner,
man_vta,
man_1_vta,
d_vta_1,
d_vta_2,
d_vta_3,
arboles,
pilas,
reglas,
seguridad,
man_arch,
tiempos,
man_imp,
man_rep,
auto_rep,
inicios,
abrcion,
menus,
morfemas

Rutinas de esta unidad:

Procedimiento : arma_semejantes
Parámetros : var palabra_fuente : string
var nombre_árbol : ap_nodo
var cantidad_nodos : integer

Descripción:
Esta rutina se encarga de crear un árbol en el que se almacenarán las palabras semejantes a una palabra de prueba, la cual se especifica a través del parámetro "palabra_fuente". El nombre del árbol que contendrá a las palabras semejantes se especifica con el parámetro "nombre_arbol". Además, esta rutina genera un resultado, el cual es el número de nodos que forman parte del árbol. El número de nodos se almacena en el parámetro "cantidad_nodos".

Ejemplo :
arma_semejantes(pal_clase,arb_semejantes,ctd_sem);

Con este ejemplo, se indica que debe construirse un árbol de palabras semejantes a la palabra contenida en la variable "pal_clase". El árbol que se construirá se identifica como "arb_semejantes". La cantidad de palabras que se encuentren como semejantes a "pal_clase" la

almacenara la rutina en la variable "ctd_semejantes".

Procedimiento : lee_dato
Descripción:
Esta rutina se encarga de leer del teclado las palabras que formarán parte de la descripción del dato a nombrar.

Procedimiento : une_texto
Parámetros : var estado_union : integer

Descripción:
Esta rutina se encarga de concatenar la última palabra escrita con el resto del texto de descripción del dato a nombrar. Asimismo, llama a la rutina "push_stack" para almacenar a dicha palabra en la pila de palabras junto con el estado del autómata del lenguaje de nomenclatura en el que se escribió dicha palabra.

Ejemplo : une_texto(edo);

Función : semantica : integer
Parámetros : tipo_elemento : tipo_palabra
palabra_prueba : cadena

Descripción:
Esta función se encarga de validar la semántica de las palabras en la descripción del dato a nombrar.
El parámetro "tipo_elemento" indica el tipo de elemento que se espera que sea la palabra a analizar.
El parámetro "palabra_prueba" representa la palabra a analizar.
Como resultado, esta función genera un código numérico que representa el valor semántico de la palabra analizada. En caso de detectar un error, el código indicará el tipo de error. Este código será entonces utilizado como información para el autómata de corrección de errores del sistema.

Ejemplo :
resultado:=semantica(art_sust,'teléfono');

En este ejemplo, se analiza que la palabra "teléfono" corresponda a un artículo o un sustantivo. Como resultado, esta función generará el código correspondiente a "sustantivo" y que se almacenará en la variable "resultado".

Procedimiento : autom_error
Descripción:

Esta rutina se encarga de controlar al autómata usado para indicar errores detectados mientras se escriben las palabras de la descripción del dato a nombrar.

Asimismo, este autómata no sólo se encarga de indicar el tipo de error detectado en la función "semántica", sino que contiene una serie de estados en los que se contemplan una serie de ayudas para el usuario que se sirvan para corregir el error y escribir correctamente la descripción del dato a nombrar.

Procedimiento : ini_errores
Descripción:
Esta rutina se usa para inicializar a la matriz de transición de estados que pertenece al autómata de detección y corrección de errores.

Procedimiento : autom_sintaxis
Descripción:
Esta rutina se encarga de controlar al autómata que ayuda a la aplicación de la sintaxis del lenguaje de nomenclatura de datos.

Procedimiento : inicia_sintaxis
Descripción:
Esta rutina se encarga de inicializar a la matriz de transición de estados que se usa en el autómata que ayuda a la aplicación de la sintaxis del lenguaje de nomenclatura de datos.

Procedimiento : inicia_listas
Descripción:
Esta rutina se encarga de inicializar a las listas de artículos y preposiciones válidas en el sistema.

Procedimiento : limpia_pila
Descripción:
Esta rutina se encarga de limpiar la pila de palabras que forman parte de la descripción del dato a nombrar.

6.- CONCLUSIONES.

La automatización de los estándares como una de las funciones de administración de datos, definitivamente es un factor crítico del éxito de la misma en la institución. En este sentido, es fundamental contar con herramientas que apoyen la definición, difusión y aplicación de los estándares de administración de datos, especialmente el de nomenclatura de datos.

Se ha mencionado que la administración de datos es un paso más en la constante evolución del área de sistemas. Es necesario que esta evolución sea claramente especificada al momento de tomar la decisión de implantar la administración de datos en la institución. Lo anterior implica no sólo la definición de funciones y procedimientos, sino trabajar en el cambio de mentalidad del personal de sistemas, así como con una fuerte capacitación en conceptos de administración de datos, administración de bases de datos, técnicas de modelado de información, técnicas de análisis y diseño de sistemas.

La definición del lenguaje de nomenclatura de datos permitió la construcción de un sistema que permite su aplicación para tener definiciones de datos correctamente estructuradas. Estas definiciones sirven para generar posteriormente los nombres de los datos.

Ahora bien, en la definición del lenguaje, se consideró

que las palabras válidas, tales como sustantivos y adjetivos, no podrían ser manejadas a través de un gran diccionario de palabras. La solución consistió en utilizar un análisis de los grafemas característicos de los sustantivos y adjetivos, de forma tal que sólo se manejasen en un diccionario a aquellas palabras que constituyeran excepciones a los grafemas. En realidad, se está proponiendo un análisis complementario al análisis sintáctico realizado por el autómata principal del sistema. Este análisis complementario, denominado **análisis morfológico**, abre las puertas para nuevas investigaciones en la teoría de lenguajes formales.

7.- GLOSARIO DE TERMINOS:

Abreviatura, abreviación :

Representación de palabras con sólo varias de sus letras.

Abreviatura convenida o por convención :

Abreviatura que es aceptada por la mayoría de los elementos de una institución, y que no corresponde a una abreviatura generada a partir de la aplicación de ciertas reglas de abreviación.

Abreviatura por construcción :

Abreviatura generada a partir de la aplicación de ciertas reglas de abreviación.

Administración de datos :

Disciplina encargada de mejorar la calidad de la información a través de la planeación, obtención, clasificación, almacenamiento, control y seguridad de los datos de una institución.

Administrador de datos :

Responsable de la administración de datos. Encargado del diseño lógico de bases de datos, nomenclatura de datos, definición de estándares de datos y planeación estratégica a través de herramientas tales como el DD/DS.

Administración de bases de datos :

Disciplina encargada del diseño físico y mantenimiento de bases de datos, así como del manejo del DBMS.

Administrador de bases de datos :

Responsable de la administración de bases de datos. Encargado de definir esquemas, subesquemas, eficiencia y acceso a las bases de datos.

Alias :

Nombres no oficiales para un mismo dato.

Ambiente de programación :

Software que apoya a la programación de sistemas de aplicación. Por ejemplo: Lenguajes de programación de tercera generación: ALGOL, Pascal, Cobol, etc. Lenguajes de programación de cuarta generación y generadores de aplicaciones como: XGEN, LINC II, etc.

Análisis de datos :

Proceso que se encarga de identificar la naturaleza de los datos de la institución. El análisis de datos se compone de: Análisis de entidades y Análisis funcional. El análisis de entidades se encarga de proveer un entendimiento sobre entidades, atributos y relaciones existentes entre éstas. El análisis funcional se encarga de proveer un entendimiento sobre las principales actividades del negocio de la institución, y de los principales datos que se requieren en ellas.

Area del negocio :

División organizacional de la institución, encargada de realizar tareas específicas del negocio. En una institución bancaria, las áreas del negocio corresponden a las bancas en las que se divide. P.ej. Banca Comercial, Banca Institucional, Banca Internacional, etc.

Arquitectura de datos :

Es una vista global de la información de la institución. Representa a las entidades, relaciones entre éstas y y los procesos en los que se requiere dicha información. Esta arquitectura de información es la base para futuros análisis detallados y construcción de sistemas. La arquitectura de información identifica las necesidades actuales y futuras de información de la institución.

Arquitectura de sistemas :

Vista global de los sistemas que se desarrollarán en la institución. La arquitectura de sistemas indica las requerimientos actuales y futuros de equipos de cómputo,

así como los sistemas de software necesarios para dar soporte a las necesidades de información de la institución.

Base de datos temática :

Conjunto de datos interrelacionados entre si, organizados en base a algún tema especifico y almacenados en forma independiente a los sistemas que hacen uso de ella.

B.S.P. Business System Planning :

Metologia de planeación estrategica de sistemas, la cual a través de una serie de matrices, identifica principales entidades, sistemas, actividades, procesos y problemas de información caracteristicos de una institución. Su propósito consiste en identificar los requerimientos globales de información de la institución, y que son necesarios para desarrollar a los sistemas de información.

Conversión lógico a físico :

Proceso mediante el cual, el diseño lógico de una base de datos es transformado a un diseño físico, el cual considera las facilidades y restricciones del DBMS en el que habrá de implementarse la base de datos.

C.S.F. Critical Success Factors :

Metodologia de planeación estratégica de sistemas que parte de la base de que las necesidades de información de la Dirección y alta gerencia pueden identificarse a partir de ciertos factores criticos en áreas claves de la institución.

Dato :

Unidad básica de información que puede identificarse y describirse en un dicccionario de datos.

Dato corporativo :

Dato que puede identificarse como común a diferentes áreas de negocio de la institución.

Descripción física del dato :

La descripción física del dato corresponde a la declaración, en un ambiente de programación, del tipo, y longitud de las variables y campos que representan al dato.

Definición del dato :

Texto que describe al dato en términos de su significado y función en el negocio de la institución. Para generar el nombre primario y nombres de programación, la definición del dato debe proveer información mínima que indique la naturaleza del dato (palabra clase) y a qué o quién hace referencia el dato (palabra entidad).

Diccionario de datos o DD/DS :

Conjunto de información organizada sobre los esquemas y subesquemas de bases de datos, archivos, campos y datos, así como de los sistemas que tienen acceso a ellos. DD/DS corresponde a las siglas para hacer referencia al software encargado de manejar a esta información.

Diseño físico de bases de datos :

El diseño físico de bases de datos consiste en la elección de los tipos de estructuras de datos a usarse para construir los archivos de una base de datos. En esta elección se consideran las consultas típicas que habrán de hacerse en la base de datos, de forma tal que las estructuras escogidas ofrezcan una ejecución eficiente en la base de datos. En esta elección deben considerarse a los parámetros propios del DBMS en el que se implantará la base de datos: tamaño de los buffers, tamaño de bloques, áreas en disco, etc.

Diseño lógico de bases de datos :

El diseño lógico de bases de datos consiste en la identificación de los requerimientos de información y en base a ellos, definir un **modelo de datos** que les dé soporte. Asimismo, el diseño lógico también define los esquemas y subesquemas de la base de datos, en base a las vistas de usuarios que deben satisfacerse. Este diseño lógico es totalmente independiente del DBMS y equipo de cómputo en el que se implantará la base de datos.

E.L.K.A. Entity Link Key Attribute :

Técnica de modelado de información, que presenta en forma gráfica a las entidades y a relaciones existentes entre éstas, las cuales forman parte de un modelo de información o modelo de datos.

Entidad :

Objeto u elemento (real o abstracto) que tiene un significado especial para el negocio de la institución, y sobre lo cual se dea almacenar datos. Ej.: Cliente, empleado, oficina, cuenta.

Esquema de la base de datos :

Definición lógica de la totalidad de una base de datos. Esta definición incluye entidades, relaciones y atributos, así como los usuarios que tienen acceso a dicha información. En el nivel físico del DBMS, el esquema corresponde a la declaración total de las estructuras físicas de datos que conforman a la base de datos.

Estándares de datos :

Estándares que rigen la definición de las características físicas de los datos, como pueden ser: tipo, longitud y valores válidos asociados a determinado dato.

Estándares de nomenclatura de datos :

Conjunto de estándares que rigen la forma de construir los nombres primarios y de programación de los datos.

Gramema :

Unidad mínima con significado en el lenguaje español. Los gramemas constituyen un conjunto reducido y finito, y su presencia en las palabras determinan su categoría gramatical. Los gramemas indican si una palabra es un sustantivo o un adjetivo, además de mostrar su género (masculino o femenino) y número (singular o plural).

Gramema de género :

Los gramemas de género clasifican a los sustantivos y adjetivos en masculinos y femeninos. Estos gramemas característicos son:
Para masculino: **o, e** .P.ej.: arquitecto, conde.
Para femenino: **a, ina, triz, esa, isa.** P.ej.: arquitectura, marquesina, actriz, condesa.

Gramema de número :

Los gramemas de número clasifican a los sustantivos y adjetivos en singulares y plurales. Estos gramemas característicos son:
Para plural: **s ó es.** P.ej.: flores, hadas.
Para singular, no se presentan los 2 gramemas anteriores. P.ej.: flor, hada.

Homónimos :

Diferentes datos que tienen asociado un mismo nombre.

Independencia de datos :

Grado de inmunidad o independencia de los programas a los cambios que se hagan a la organización lógica, organización física y estrategias de acceso usadas en bases de datos.

Integridad de datos :

Término utilizado para indicar el estado correcto de una base de datos. Este estado correcto se mide en términos de la exactitud y consistencia de los datos al momento de actualizar a la base de datos. La integridad de datos también se refiere a la característica de obtener los resultados esperados en base a actualizaciones típicas en la base de datos; si en base a estas actualizaciones se obtienen resultados inesperados en los valores de los datos, se dice que la base de datos ha perdido su integridad.

Lenguaje de nomenclatura de datos :

Lenguaje formal, inspirado en el **lenguaje OF** y con características del lenguaje español, que tiene el propósito de permitir al administrador de datos estructurar correctamente las definiciones de los datos (en base a palabras clase, entidad y modificadores) para aplicar el proceso de nomenclatura de datos.

Lenguaje OF :

Lenguaje desarrollado por I.B.M., a modo de guía, para estructurar las definiciones de los datos en base a palabras clase, palabras entidad y modificadores.

Metadatos :

Datos acerca de datos, es decir, la definición y descripción del recurso de los datos, sus características, nombres (primario y de programación), localización, usos, etc. Los metadatos se usan para identificar, describir, definir y nombrar a los datos de la institución. El contenido de un diccionario de datos son los metadatos.

Modelado de datos :

Actividad encargada de analizar los procesos del negocio en base a la información que en ellos se requiere y genera, y de esta forma identificar entidades y relaciones así como los atributos de las entidades. Los productos de esta actividad son los modelos de información y los modelos de datos de la institución.

Modelo de datos :

Es una representación que está compuesta por diagramas que muestran a las entidades, sus relaciones e identificadores lógicos en una forma mucho más detallada que el modelo de información. El modelo de datos debe cumplir con el proceso de normalización de bases de datos. Además de los diagramas de entidades y relaciones, un modelo de datos incluye a una serie de reglas que permitan asegurar la integridad de los datos.
Entre los modelos de datos y los modelos de información debe existir una clara correspondencia entre ellos.

Modelo de información :

Es una representación en forma de diagrama que muestra a entidades, sus relaciones y así como sus identificadores lógicos. Su propósito consiste en mostrar en forma clara y a un nivel conceptual a la información requerida en los procesos del negocio de la institución.

Nombres de programación :

Son los nombres de los campos de archivos (nivel de lenguajes de programación) que se usan para representar a los datos.

Nombre primario :

Es la identificación a nivel lógico del dato. Se usa como llave en el diccionario de datos para tener acceso a los metadatos del dato.

Nomenclatura de datos :

Proceso de la administración de datos que se encarga de crear una definición única para cada dato de la institución. A partir de dichas definiciones se nombrarán a los datos y serán puestos a disposición del personal de sistemas y usuarios finales.

Normalización de bases de datos :

Proceso en el diseño de bases de datos que tiene como propósito el descomponer ciertas entidades y relaciones de forma tal que no se presenten anomalías en la inserción, supresión y actualización de registros de la base de datos.

Palabra clase :

Sustantivo que describe en forma explícita al dato de acuerdo a su uso en los procesos del negocio. Específicamente, la palabra clase brinda informarmación que contesta a la pregunta ¿Qué es?.
Ej: nombre, número, fecha, porcentaje .

Palabra clase de modificador forzoso :

Tipo de palabra clase cuyo significado no brinda información completa sobre la naturaleza del dato, por lo que requiere la presencia de palabras modificadoras en la definición y nombres del dato. P.ej: Fecha.

Palabra clase de modificador opcional :

Tipo de palabra clase cuyo significado brinda información completa sobre la naturaleza del dato, por lo que no requiere la presencia de palabras modificadoras en la definición y nombres del dato.

Palabra entidad :

Sustantivo que identifica a los sujetos del negocio acerca de los cuales se desea almacenar información. La palabra entidad contesta a la pregunta ¿De quién?. Estas palabras entidad deben corresponder a las entidades identificadas en los modelos de información y modelos de datos corporativos y de las áreas del negocio.

Palabra modificadora :

Son palabras (sustantivos y adjetivos) que complementan la definición del dato. Mediante estas palabras modificadoras se hace que la definición del dato pueda tener un significado más claro. Dependiendo del tipo de la palabra clase usada en la definición del dato a nombrar, la presencia de un modificador sera forzosa u opcional.

Planeación estratégica de sistemas :

Actividad encargada de producir una arquitectura de información que refleje las fronteras, interrelaciones, orígenes y flujos de los diversos sistemas de información y de esta forma poder tener una guia priorizada de su desarrollo y mantenimiento sistemático.

Redundancia de datos :

Almacenamiento de datos en forma repetida en diferentes archivos. Esta redundancia se dice que es nociva cuando no se tiene un control en la forma que se actualizarán los datos para garantizar que el valor de un dato coincidirá en el mismo momento en los diferentes archivos en los que se encuentra almacenado.

Reglas de abreviación :

Conjunto de reglas utilizadas para generar la abreviatura de las palabras clase, entidad y modificadores en el proceso de nomenclatura de datos.

Reglas de integridad :

Conjunto de especificaciones que indican la forma de conservar un alto grado de consistencia y exactitud de los datos almacenados en una base de datos.

Sinónimo :

Nombres oficiales que se asocian a un mismo dato.

Subesquema de la base de datos :

Definición parcial de la base de datos. Un subesquema da soporte a las necesidades de información detectadas en la correspondiente vista de usuario.

Vista de usuario :

Porción de información que el usuario requiere para llevar a cabo las actividades a las cuales está relacionado en su trabajo.

BIBLIOGRAFIA:

1. Aho, Alfred V. and J.D. Ullman, **Principles of Compiler Design.** Addison-Wesley, Reading, Mass., 1977.

2. Brathwaite Kenneth S. **Systems Design in a Database Environment.** Intertext Publications. McGraw-Hill Book Company. 1989.

3. Cárdenas Alfonso F. **Data Base Management Systems.** 2nd Edition. Allyin and Bacon. 1985.

4. Durell William. **Data Administration.** McGraw-Hill Book Company. 1988

5. Hopcroft, J. E. and J. D. Ullman, **Formal Languages and Their Relation to Automata.** Addison-Wesley, Reading, Mass., 1977.

6. Kolman Bernard and Busby Robert, **Estructuras de Matemáticas Discretas para la Computacion.** Prentice Hall. 1986.

7. López Miguel, Casanova Francisco, Menendez Rafael, **Español Moderno I.** Ed. Progreso. 1978

8. López Miguel, Casanova Francisco, Menendez Rafael, **Español Moderno II.** Ed. Progreso. 1978

9. Plagman Bernard and Leong-Hong Belkis, **Data Dictionary/Directory Systems. Administration, Implementation and Usage.** John Wiley & Sons Inc. 1982.

10. Standish Tomas A. **Data Structures Techniques.** Addison Wesley, 1980.

BIBLIOGRAFIA:

1. Aho, Alfred V. and J.D. Ullman, **Principles of Compiler Design**. Addison-Wesley, Reading, Mass., 1977.

2. Brathwaite Kenneth S. **Systems Design in a Database Environment.** Intertext Publications. McGraw-Hill Book Company. 1989.

3. Cárdenas Alfonso F. **Data Base Management Systems.** 2nd Edition. Allyin and Bacon. 1985.

4. Durell William. **Data Administration.** McGraw-Hill Book Company. 1988

5. Hopcroft, J. E. and J. D. Ullman, **Formal Languages and Their Relation to Automata.** Addison-Wesley, Reading, Mass., 1977.

6. Kolman Bernard and Busby Robert, **Estructuras de Matemáticas Discretas para la Computacion.** Prentice Hall. 1986.

7. López Miguel, Casanova Francisco, Menendez Rafael, **Español Moderno I.** Ed. Progreso. 1978

8. López Miguel, Casanova Francisco, Menendez Rafael, **Español Moderno II.** Ed. Progreso. 1978

9. Plagman Bernard and Leong-Hong Belkis, **Data Dictionary/Directory Systems. Administration, Implementation and Usage.** John Wiley & Sons Inc. 1982.

10. Standish Tomas A. **Data Structures Techniques.** Addison Wesley, 1980.