UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Superiores Acatlán

Programación de plantillas para documentos técnicos utilizando LaTeX

Tesis presentada por

Asael Fabian Martínez Martínez

para obtener el título de

Licenciado en Matemáticas Aplicadas y Computación

Asesora

Mtra. MariCarmen González Videgaray

Naucalpan, Estado de México Abril del 2006

Resumen

El propósito de este trabajo es proporcionar cinco clases programadas en LaTeX 2ε para elaborar documentos técnicos. Se comienza con una explicación acerca de los documentos técnicos: sus características, la metodología que se debe seguir para elaborarlos y se profundiza en la explicación de los documentos técnicos que se utilizan con frecuencia en la carrera de Matemáticas Aplicadas y Computación: documentación de software, manuales de usuario, presentaciones y trabajos de investigación y titulación. Asimismo, se explica ampliamente cómo realizar documentos técnicos con el lenguaje LaTeX 2ε, un lenguaje de marcado. Las plantillas son indispensables para agilizar la creación de documentos técnicos; en este caso se programan cinco clases en LaTeX 2ε para los documentos técnicos mencionados anteriormente. Por último, se da una explicación de cómo utilizar cada una de estas plantillas.

Palabras clave: documentación técnica, LaTeX 2ε, plantillas, programación en LaTeX

Abstract

The purpose of this work is to provide five classes programmed in LaTeX 2ε to make technical writing easier. It starts with an explanation of technical writing: its characteristics, methodology and a review of the most frequently used documents in the field of Mathematics and Computation: software documentation, user manuals, slides, reports and thesis. There is also an explanation of the use of LaTeX 2ε as a markup language in order to develop technical documents. As templates are essential to speed up technical writing, this work offers five classes in LaTeX 2ε. Finally, the present document shows how to use each template.

Keywords: technical writing, LaTeX 2ε, templates, LaTeX programming

Agradecimientos

Alguien dijo alguna vez «todo tiene su tiempo» y, nos demos cuenta o no, así es. Después de algunos años de estudio he llegado a concluir una etapa más de mi vida, en esta ocasión en el área académica. Y como el título de esta sección dice, quiero agradecer a quienes han estado cerca de mi durante este tiempo.

En primer lugar quiero dar gracias a mis padres: Manuel y Mercedes. En estos años de estudio siempre han confiado en mi y en lo que puedo hacer, me han dado su apoyo incondicionalmente, y, si me voy hasta el principio de mis días, gracias a ellos estoy aquí. Gracias por ser mis padres y hacer de mi lo que ahora todos pueden ver.

También quiero agradecer a todos mis maestros, aquellos que de una forma u otra han contribuido en mi formación. Alguna vez escuché que no es lo mismo un profesor que un maestro, y estoy de acuerdo; un maestro va más allá que un profesor. El maestro en todo momento trata de enseñarte algo, por pequeño que sea, para hacer de ti una mejor persona (además de que algunos contribuyen en tu formación académica); a esas personas son a las que quiero darles las gracias.

De igual manera, quiero dar gracias a todos mis compañeros con quienes pude pasar momentos de todo tipo. A los que se llegaron a convertir en amigos, les agradezco también esa confianza, y sé que puedo contar con ellos en cualquier momento.

Quienes me conocen, saben que no soy de muchas palabras, así que en general quiero agradecer a todas aquellas personas de las cuales he aprendido lo que quiero ser y lo que no quiero ser en esta vida.

Por último (aunque siempre es el primero), Abba, gracias por haber creado todas las cosas.

A SELECT z
FROM אֶרֶץ, שָׁמַיִם
WHERE z IN ♡

אָבִינוּ שֶׁבַּשָּׁמַיִם יִתְקַדַּשׁ שְׁמֶךָ
תָּבֹא מַלְכוּתֶךָ יֵעָשֶׂה רְצוֹנְךָ בָּאָרֶץ כַּאֲשֶׁר נַעֲשָׂה בַשָּׁמָיִם
תֵּן־לָנוּ הַיּוֹם לֶחֶם חֻקֵּנוּ
וּסְלַח־לָנוּ אֶת־אַשְׁמָתֵנוּ כַּאֲשֶׁר סָלְחִים אֲנַחְנוּ לַאֲשֶׁר אָשְׁמוּ לָנוּ
וְאַל־תְּבִיאֵנוּ לִידֵי מַסָּה כִּי אִם־הַצִּילֵנוּ מִן־הָרָע אָמֵן

מַתָּי 13-9.6

Índice general

Introducción

Todos alguna vez hemos visto alguna película donde un avión es secuestrado por un grupo de terroristas. De alguna manera, el héroe de la película logra terminar con los malos, pero durante la batalla, los pilotos del avión son asesinados. Ahora el avión está en manos del héroe, que conoce de todo, excepto cómo aterrizar un avión. Es entonces cuando nuestro héroe le pide a la azafata que busque en el «Manual de Vuelo» la sección donde se dan las instrucciones de aterrizaje. De esta manera el héroe logra aterrizar el avión sin muchos problemas, salvando así a todos los pasajeros.

¿Qué hubiera pasado si el Manual de Vuelo tuviera algún error, estuviera redactado de una manera confusa o no fuera posible localizar la sección de aterrizaje rápidamente? Seguramente todos los pasajeros habrían muerto, eso sin contar los daños que hubiera ocasionado el avión en el lugar donde se estrellara.

Tal vez los manuales de vuelo no sean muy utilizados por un estudiante de Matemáticas Aplicadas o de Computación, pero los problemas originados por una mala «documentación» no son tan triviales. Un ejemplo claro es la «documentación del software». El primer problema al que se enfrenta cualquier programador cuando tiene que dar mantenimiento a algún programa de cómputo es entender el código fuente, ya sea porque él no lo escribió o porque no recuerda con claridad qué programó; en ocasiones es más fácil volver a hacer el programa que corregirlo.

Los manuales de vuelo (y en general cualquier tipo de manual), la documentación de software y otros documentos que contienen información sobre un tema específico se conocen como «documentos técnicos». Los documentos técnicos, a pesar de su nombre, son utilizados con frecuencia por casi toda la gente. Por ejemplo, los «manuales de usuario» se utilizan para instruir sobre cómo utilizar algún producto; en las escuelas se utilizan «libros de texto» para enseñar a los alumnos alguna materia específica, ya sea a nivel básico, como historia o español, o a nivel licenciatura, como cálculo o medicina forense; y actualmente las «páginas web» son utilizadas en todo el mundo para dar a conocer información sobre algún tema específico.

La «documentación técnica» es la encargada de que los documentos técnicos mencionados anteriormente, y otros más, se elaboren de manera correcta, cumpliendo así su objetivo: que el lector lea el documento y lo entienda.

Actualmente, es frecuente observar que cada disciplina tiene un lenguaje propio, el cual en ocasiones no es fácil entender, y más aun, cuando personas de distintas disciplinas interactúan, la comunicación puede ser difícil; sin embargo, la documentación técnica quita esas barreras, ya que los documentos que se elaboran deben contener información clara para cualquier persona.

Una herramienta muy importante en la documentación técnica es la computadora. Los procesadores de texto permiten capturar y dar formato a los documentos. Actualmente hay una gran varie-

dad de procesadores de texto en el mercado, el más conocido es MS-WORD, pero existen otros, por ejemplo: FRAMEMAKER, VENTURA PUBLISHER, FREEHAND, TEX y LATEX, cada uno de ellos tiene sus propias características.

TEX, el antecesor de LATEX, es un lenguaje de programación que está orientado a la creación de documentos técnicos y científicos. TEX y LATEX han sido aceptados en muchas universidades y editoriales, debido a la calidad con que componen los documentos, principalmente aquellos que contienen notación matemática.

LATEX se considera un lenguaje de marcado, como lo es HTML. Un lenguaje de marcado funciona a través de «etiquetas», las cuales indican el formato que se le deberá dar a un texto. Esta forma de crear documentos ayuda a los escritores a enfocar su atención en la información que se desea transmitir, en lugar de ocuparse de la forma en que se presentará dicha información. Por ejemplo, si se desea iniciar un capítulo, en un lenguaje de marcado se tendría que escribir lo siguiente:

```
<capítulo>Documentación técnica</capítulo>
```

y el intérprete o compilador sabrá qué formato darle al título; mientras que en un procesador de textos es necesario que el escritor sea quien dé el formato, es decir, tendría que cambiar esa parte del texto a una tipografía de 22 puntos, sans-serif y negrita, además de que se deberá colocar este título en una página nueva.

Un elemento importante dentro de LATEX son las «clases» (o «plantillas», en términos de la documentación técnica). Una clase es un archivo que contiene el formato del documento, es decir, en este archivo se define el tamaño de papel, los márgenes y la tipografía a utilizar; además contiene la definición de etiquetas (conocidas en LATEX como «instrucciones» y «entornos») las cuales permiten crear elementos especiales dentro del documento, por ejemplo, una tabla o el título de un capítulo. LATEX también proporciona diversas clases para crear distintos tipos de documentos, por ejemplo, libros, trabajos científicos o técnicos, presentaciones, etc.; además, existe gran variedad de clases que fueron creadas según las normas de editoriales o universidades.

En este trabajo se utilizará LATEX para crear documentos técnicos. Se eligió LATEX por las siguientes razones:

- Es un lenguaje de programación. Todos aquellos que han recibido una formación matemática, tienen la facilidad para aprender a utilizar lenguajes de programación. Como se mencionó, LATEX es un lenguaje de marcado, así que aprendiendo la sintaxis y unas pocas instrucciones, será posible elaborar un documento.
- Los documentos creados con LATEX son de la más alta calidad. Desde que TEX apareció, fue bien recibido por quienes tenían que escribir libros (en primer lugar quienes escribían libros sobre matemáticas, pero no se limita a ellos), ya que los documentos que compone tienen un diseño muy bueno. LATEX, a pesar de ser más simple que TEX, no pierde para nada esa calidad.
- Es un estándar internacional. Muchas universidades, instituciones y editoriales han adoptado a TEX y a LATEX como estándares para sus publicaciones. Por ejemplo, la IEEE recomienda a quienes les desean mandar algún documento, utilizar estos lenguajes si su trabajo contiene matemáticas, incluso esta institución ha creado plantillas para sus publicaciones. De igual manera, la AMS recomienda fuertemente el uso de LATEX a todos aquellos que deseen escribir un artículo o libro para ellos.

- Quien utiliza LaTeX dedica más tiempo al contenido del documento que al diseño del mismo. Todos aquellos que alguna vez han utilizado un procesador de textos, frecuentemente ocupan un tiempo considerable en «diseñar» su documento, es decir, tienen que elegir el tipo de letra y el tamaño tanto para el texto normal como para los distintos encabezados, definir márgenes, sangrado, interlineado y la colocación del número de página entre otras cosas más. En cambio, con LaTeX todo lo anterior queda en manos precisamente de LaTeX y de la plantilla; de esta forma, el escritor podrá dedicarle más tiempo al contenido del trabajo (lo que realmente importa) que al diseño.
- El usuario debe ser organizado. Cualquiera que ha tenido que hacer un programa, sabe que primero necesita tener bien definido lo que va a hacer, es decir, necesita de un algoritmo o pseudocódigo, de lo contrario, el programa será un desastre. Lo mismo sucede con LaTeX. Al ser un lenguaje de programación, LaTeX requiere de mentes organizadas (en este caso el «algoritmo» o «pseudocódigo» se obtiene siguiendo la metodología de la documentación técnica) de lo contrario el código fuente del documento puede llegar a ser confuso, incluso el propio LaTeX.
- Es portable. Actualmente es común encontrar computadoras con sistemas LINUX, MAC-OS, WINDOWS, etc. y uno de los problemas es que los programas de cómputo no funcionan en todas las plataformas. Pero LaTeX es la excepción. Existen versiones de LaTeX para casi todas las plataformas, y gracias al formato de sus archivos fuente (texto plano), es posible crear un documento en un sistema LINUX y modificarlo en un sistema WINDOWS, por ejemplo, sin que la apariencia y calidad del documento final se vea afectada. Además, uno de los formatos que maneja LaTeX para crear los documentos finales es el `pdf`, el cual es el más utilizado por quienes utilizan internet.

A pesar de todo lo anterior, LaTeX casi no se conoce en nuestro país. Pocas son las instituciones que lo conocen y menos las que lo difunden. Específicamente, en la U.N.A.M. únicamente existe una plantilla para crear presentaciones con LaTeX.

El objetivo de este trabajo es proporcionar una serie de plantillas programadas en LaTeX para elaborar los documentos técnicos más utilizados tanto en la carrera de Matemáticas Aplicadas y Computación como en áreas afines, tanto profesionales como académicas. Además de lo anterior, este trabajo proporciona los lineamientos para elaborar un documento técnico correctamente, especialmente (como se mencionó) los que se utilizan frecuentemente en la carrera; de igual manera, se indica cómo utilizar el lenguaje LaTeX 2_ε como herramienta para crear dichos documentos.

El capítulo 1 de este trabajo trata todo lo relacionado con la documentación técnica mencionada anteriormente. Se da una explicación de qué es un documento técnico, qué características debe tener el documento, cuál es su estructura general y cuál su clasificación. Además, en este capítulo se habla brevemente del «mapeo de información»; que es una metodología que ayuda a organizar y distribuir la información de un documento para que sea fácil de leer y de entender. Se incluye también una sección dedicada a la bibliografía.

Para crear correctamente un documento técnico se debe seguir cierta metodología. Esta metodología se explica también en el primer capítulo. Por último, se tratan de manera más profunda cinco de los tipos de documentos técnicos que se utilizan en la carrera de Matemáticas Aplicadas y Computación, estos son: documentación de software, manuales de usuario, presentaciones, trabajos de investigación y trabajos de titulación. En cada uno de estos documentos se explica cómo se genera la documentación y cuál es su estructura general.

No se pretende dar una amplia explicación de cada uno de los elementos que componen un documento, ni se profundiza en la gramática u ortografía, simplemente se dan lineamientos para facilitar la elaboración de los documentos que se utilizan con más frecuencia.

El capítulo 2 está dedicado a estudiar más a detalle qué es LaTeX y cómo funciona. Como se ha mencionado, LaTeX es un lenguaje de marcado orientado a la creación de documentos de alta calidad, principalmente aquellos que utilizan notación matemática. En este capítulo se explica cómo crear documentos, se comienza por la sintaxis de LaTeX y se explica cómo componer elementos comunes de un documento, por ejemplo: encabezados de títulos, listas, tablas, referencias cruzadas, notación matemática, bibliografía e índices; se menciona también cómo crear presentaciones con la clase prosper.

Existen dos versiones de LaTeX que se utilizan actualmente: LaTeX 2.09 y LaTeX 2_ε, la primera se considera obsoleta, mientras que la segunda es la versión utilizada actualmente. En este trabajo se utilizará LaTeX 2_ε.

La instalación de LaTeX, si bien no es prioridad en este trabajo, es conveniente que pueda realizarse fácilmente. El apéndice A contiene una explicación de cómo instalar LaTeX y sus herramientas complementarias, tanto para sistemas WINDOWS como para sistemas LINUX. También se da una explicación breve de cómo convertir una presentación creada con la clase prosper de formato PostScript a PDF.

El capítulo 3 está relacionado con la programación de plantillas con LaTeX. Aquí se explica qué es programar en LaTeX y cómo se realiza.

Si bien LaTeX está orientado a crear documentos, internamente maneja instrucciones propias de un lenguaje de programación, como son condicionales y ciclos, lo cual aumenta su capacidad para crear elementos especiales, además de que permite crear el equivalente a funciones en los demás lenguajes: instrucciones y entornos. Precisamente utilizando todos estos elementos se crean las plantillas.

Durante la creación de un documento técnico se elabora una «guía de estilos» (esto se menciona en el capítulo 1); con LaTeX, la forma de crear esta guía de estilos es precisamente por medio de la creación de instrucciones y entornos.

En cuanto a la forma de programar, los expertos en LaTeX recomiendan utilizar la «programación literaria». La programación literaria consiste en mezclar el código de la plantilla con su explicación, de esta forma se entiende fácilmente qué se está haciendo. También es recomendación de estos expertos que se utilice la versión LaTeX 2_ε, ya que es la única que será considerada para futuras versiones de LaTeX.

Este capítulo termina con una explicación de cómo se creará cada una de las plantillas para documentos técnicos mencionados en el capítulo 1.

Por último, el capítulo 4 muestra los resultados que se obtienen utilizando cada una de las plantillas del capítulo 3. Se explican los elementos especiales de cada plantilla, cómo se utilizan y qué hacen. Por razones de espacio no se incluyen los documentos completos en este trabajo, únicamente se muestran algunas páginas de estos; para consultar los documentos completos se puede revisar el CD.

Debido al tipo de trabajo, se anexa un CD el cual contiene, entre otras cosas, las plantillas elaboradas y los ejemplos del capítulo 4 completos, tanto el código fuente como el resultado final. El apéndice B muestra el contenido de dicho CD.

Capítulo 1

Documentación técnica

La documentación técnica es la rama de la comunicación que se encarga de presentar la información de manera adecuada, de tal forma que pueda ser leída y comprendida fácilmente por los lectores. La documentación técnica tiene sus inicios en la milicia, con la documentación militar e información clasificada, y ha llegado hasta nuestros días, donde los documentos electrónicos se han vuelto muy comunes y existen organismos internacionales que han fijado normas de estilo para crear documentos técnicos efectivos.

El objetivo de la documentación técnica es transmitir ideas, información o descubrimientos de carácter técnico o científico. Es necesario aclarar que la palabra «técnica», según el diccionario de la Real Academia de la Lengua Española, significa:

> *adj.* Dicho de una palabra o de una expresión: Empleada exclusivamente, y con sentido distinto del vulgar, en el lenguaje propio de un arte, ciencia, oficio, etc.

Así, con ayuda de la documentación técnica es posible elaborar documentos que traten, por ejemplo, de álgebra lineal, cálculo multivariado, procesos estocásticos, medicina forense, derecho civil, métodos de enseñanza, finanzas, pintura, carpintería, etc.

Dado que la documentación técnica es una disciplina muy amplia, sólo se tratarán puntos básicos de la misma, como son: los documentos técnicos (que son el producto de la documentación técnica), sus características, estructura, clasificación y la metodología que se debe seguir para crear el documento correctamente; además de que se precisará dónde entra la documentación técnica en la carrera de Matemáticas Aplicadas y Computación.

1.1. Documentos técnicos

Un documento técnico, según González Videgaray (2004), es «aquel escrito que contiene información acerca de un área de conocimiento, presentada de manera estructurada, para ser presentada eficazmente a los lectores». Algunos ejemplos de documentos técnicos son:

- Manuales de usuario
- Manuales de organización
- Libros de texto

- Artículos de revistas
- Escritos científicos
- Documentación de software
- Sitios web
- Interfaces
- Presentaciones
- Correos electrónicos

Al elaborar un documento de este tipo se pretende que el lector comprenda el contenido y pueda localizar rápidamente la información que requiera. Asimismo, el documento debe tener una estructura que permita ser actualizado fácilmente.

1.1.1. Características

Para que un documento técnico sea efectivo, su contenido debe poseer las siguientes características:

- Objetividad
- Precisión
- Claridad
- Concisión
- Variedad
- Convicción

Objetividad A diferencia de una obra literaria, donde el autor puede expresar libremente su sentir, la información proporcionada en un documento técnico debe estar fundamentada técnica o científicamente, según el tipo de trabajo.

La subjetividad es aceptable en parte, sólo en la introducción o en la discusión de los resultados del trabajo.

Precisión El vocabulario que se utilice dentro del documento no debe ser ambiguo en su significado, de modo que cualquier persona que lea el documento entienda lo mismo. De igual manera la exposición de ideas debe ser lógica.

Dicho de otra manera, las palabras deben utilizarse en el momento y lugar adecuados.

Claridad Para lograr la claridad dentro de un documento es importante conocer la clase de lectores a quienes va dirigido; esto es necesario porque se tiene que utilizar un lenguaje adecuado según sus conocimientos y capacidades.

Esto ayuda a no tener que trabajar en exceso durante la elaboración del documento, ya que se sabrá qué tan «profunda» debe ser la explicación del tema.

Concisión En una obra literaria es común utilizar gran cantidad de figuras retóricas, pero esto no es válido dentro de la documentación técnica. Es importante saber expresar las ideas utilizando la menor cantidad de palabras posibles.

Este punto está relacionado con la objetividad.

Variedad Aquí la variedad debe ser en cuanto al uso del lenguaje. La persona que escribe el documento debe conocer bien el lenguaje para estructurar las ideas.

Es importante no ser monótono en la redacción, ya que esto llega a ser aburrido. Pero tampoco se debe caer en el exceso de variedad para evitar posibles confusiones en la comprensión de la lectura.

Convicción Dentro de un documento técnico o científico se deben realizar afirmaciones con toda rotundidad y convicción que le permitan los datos disponibles.

1.1.2. Estructura

Un documento técnico consta de tres grandes partes:

- Preámbulo
- Contenido
- Referencias

Estas, a su vez, tienen se conforman de otras secciones.

Preámbulo

La primera parte de un documento es el preámbulo. En él se encuentran las siguientes secciones:

- Portada
- Introducción
- Tabla de contenido

La numeración de las páginas en el preámbulo deberá ser con números romanos (I, II, III, ...).

En la introducción se describe brevemente el contenido del documento, con el fin de que el lector pueda saber si encontrará la información que busca o no. Esta sección es distinta a la introducción que se da en un trabajo de investigación, más bien corresponde al conocido «resumen» o «abstract», y en el caso de un manual corresponde al prefacio.

Contenido

Después del preámbulo se encuentra el contenido. Aquí se expone el tema completo. De manera general se incluyen las siguientes secciones:

- Descripción del entorno
- Desarrollo

La numeración de las páginas se reinicia y se cambia a números arábigos (1, 2, 3, ...).

Las secciones del contenido dan una secuencia lógica a todo el trabajo, y pueden variar dependiendo del tipo de documento que sea. En la descripción del entorno se describe el propósito del documento; en la parte de desarrollo se expone todo el tema y, si se trata de un documento científico, se incluyen las conclusiones, explicando si se cumplió o no lo planteado en un principio.

Referencias

Esta última parte proporciona información adicional sobre el documento. De manera general se incluyen las siguientes secciones:

- Apéndices
- Fuentes
- Índices

Los apéndices se deben incluir sólo si son necesarios, en estos se proporciona información que no es tan importante para comprender el contenido principal del trabajo.

En la sección de las fuentes es necesario incluir todas las referencias que permitan profundizar sobre el tema o continuarlo. La composición de las referencias debe seguir alguna norma de estilo.

Los índices permiten localizar información específica dentro del texto. Existen varios tipos de índices, dependiendo del tipo de documento, entre ellos se encuentran: temáticos, de autores y terminológicos.

El nivel de detalle de un índice es hasta de tres niveles y en cada nivel se coloca un concepto (conocido también como entrada). Cada entrada es ordenada alfabéticamente y corresponden a los conceptos principales que son tratados en el documento y facilitan su localización mediante referencias al número o a los números de página correspondientes; en ocasiones se hace una referencia cruzada a otra entrada dentro del mismo índice.

1.1.3. Clasificación

Los documentos técnicos se pueden clasificar por su función o por su medio de distribución.

Por su función

De acuerdo a su objetivo primordial, un documento técnico se puede clasificar en seis tipos: 1) manuales, 2) reportes, 3) comunicados, 4) presentaciones, 5) ayudas en línea y 6) documentos interactivos.

Manuales En un manual se dan indicaciones precisas para formas de actuar. Un manual puede ser: de usuario, de referencia, de organización, de procedimientos o de operaciones.

Reportes Un reporte puede ser un listado, una forma, un formato o un machote. Los formatos deben ser fáciles de llenar, con una estructura lógica y bien definida. Los reportes deben estar diseñados para que se pueda localizar fácilmente la información importante.

Comunicados Los comunicados deben contener toda la información relevante, en una estructura lógica. Un comunicado puede ser un correo electrónico, un oficio, un memorándum o una circular.

Presentaciones Dentro de las presentaciones se incluyen los materiales para cursos, apoyo didáctico, exposiciones y ponencias. En una presentación, la información que muestre debe ser breve, concisa y bien estructurada.

Ayudas en línea Actualmente la documentación para uso de software se crea en archivos especiales que pueden ser utilizados en forma local o vía remota. Este tipo de documentos deben ser cortos, fragmentados, con índice y glosario.

Documentos interactivos En esta categoría entran los sitios web y formas electrónicas. Es importante no abusar de los elementos visuales, ni de animaciones o multimedia.

Por el medio de distribución

Es importante saber cómo se hará llegar un documento técnico a los usuarios, ya que esto influye en cómo diseñar el documento en cuanto al contenido y tipografía. En general, se tienen tres medios de distribución: 1) impreso, 2) electrónico y 3) proyecciones.

Los documentos impresos se hacen sobre cualquier tipo de papel, por ejemplo, un libro de texto. Los documentos electrónicos son cada vez más frecuentes, ya que es más fácil distribuirlos y ahorran papel. Por último, las proyecciones sirven para presentar el documento simultáneamente a varias personas, por ejemplo, por medio de acetatos.

1.1.4. Mapeo de información

El «mapeo de información» es una metodología que tiene como objetivo que los documentos técnicos sean fáciles de leer e inviten a la lectura. Para lograrlo, el documento debe ser sencillo en su contenido, estructura y presentación. Los principios del mapeo de información están fundamentados en la psicología de la percepción y en las ciencias cognoscitivas.

El mapeo de información consiste en dividir la información en «mapas». Con estos mapas se tendrá un documento con una estructura de un árbol, primero se presenta la información general, y luego se tienen las ramas, proporcionando información más detallada. Cada mapa debe ser conciso, de manera que pueda ser entendido fácilmente por el lector.

Principios del mapeo de información

El mapeo de información se basa en siete principios básicos de la comunicación. Dichos principios son:

1. Principio de fragmentación
2. Principio de relevancia
3. Principio de etiquetado
4. Principio de consistencia

5. Principio de gráficas integradas
6. Principio de detalle accesible
7. Principio de jerarquía de fragmentos y etiquetas

Estos principios dan como resultado documentos con un estilo de escritura que satisface las necesidades de los lectores y de los escritores. A continuación se explica cada uno de estos principios.

Principio de fragmentación La información se debe manejar en unidades pequeñas y manejables, que contengan una sola idea. Esto consiste en identificar los elementos que constituyen el documento general y separarlos de manera visible para el lector.

Principio de relevancia Toda la información contenida en un fragmento se referirá a una sola idea basada en su propósito o función para el lector. Si se tiene más de una idea en un fragmento, es posible que alguna de ellas no sea percibida correctamente.

Principio de etiquetado Una vez fragmentada y organizada la información, se debe colocar una etiqueta a cada bloque. La etiqueta consiste en un texto corto que describe el contenido de la información; ésta debe colocarse del lado izquierdo del documento.

Principio de consistencia Se deben utilizar palabras, etiquetas, formatos, organización y secuencias similares para temas semejantes. Es decir, se deben utilizar los mismos estilos y patrones en todo el documento, así como la misma terminología, notación y nomenclatura. Aplicando este principio, los lectores podrán localizar la información rápidamente, además de que no tendrán que perder tiempo en comprender la estructura del documento.

Principio de gráficas integradas Se deben usar diagramas, tablas, ilustraciones, etc., como una parte integral del texto y no como un complemento que se agrega después de terminar de escribir. Las gráficas mantienen la continuidad del texto y son parte de él.

En un documento se considera como gráfica todo aquello que no sea un texto, esto es, tablas, diagramas, ilustraciones, dibujos, etc. Las gráficas deben ser absolutamente claras, relevantes y explicar mejor que un texto. En otras palabras, sólo deben aparecer si aportan claridad al documento.

Principio del detalle accesible El documento debe estar escrito al nivel de detalle que haga que la información que el lector necesita sea accesible y que el documento sea útil para todos los lectores. En otras palabras, se debe poner lo que el lector necesita donde lo necesita. Se deben incluir visiones generales, sumarios, descripciones, glosarios, diagramas y ejemplos claramente etiquetados para presentar toda la información «abstracta».

Principio de la jerarquía de fragmentos y etiquetas Los fragmentos relevantes de información se deben agrupar en forma jerárquica y se debe poner una etiqueta a cada uno de los grupos más grandes.

1.1.5. Bibliografía

La bibliografía es indispensable en los documentos técnicos, principalmente en trabajos de investigación, ya que permiten que algún lector interesado en cierta parte del documento pueda profundizar más sobre él. Además, a lo largo del trabajo es común hacer referencia a algún documento por medio de las citas bibliográficas.

Para elaborar la bibliografía así como el texto de la cita se debe recurrir a algún estilo definido. En Turabian (1987) se muestran los distintos estilos definidos por el *Manual de Estilo Chicago*.

Sistemas de citación bibliográfica

Los sistemas de citación definen cómo se presentan las referencias hechas dentro del documento. Los sistemas más utilizados se describen a continuación:

Autor-fecha Las citas bibliográficas se forman con el apellido del autor seguido del año de la publicación, este último entre paréntesis. Cuando se trata de dos autores se incluyen ambos, por ejemplo: Pujol y Solà (1995). Si son más de dos, sólo se incluye el primero seguido de la abreviación latina «et al.», por ejemplo: Goossens et al. (1993). En caso de haber dos o más citas idénticas pero correspondientes a publicaciones diferentes, se distinguen mediante una letra al final del año de la publicación: Knuth (1986a), Knuth (1986b).

Numérico Las citas se numeran entre corchetes, por ejemplo, la primera entrada en la lista de referencias bibliográficas será: [1]. En cuanto al orden en que se numeran las citas, se tienen dos opciones: o bien según el orden de aparición dentro del trabajo, o bien ordenando alfabéticamente por autor.

Referencias insertadas Las citas bibliográficas se reemplazan por referencias bibliográficas completas. Por ejemplo: J. M. Pujol y J. Solà (*Ortotipografia. Manual de l'autor, l'autoeditor y el dissenyador gràfic*. Columna, Barcelona, 1995).

Autor-fecha abreviado Las citas se componen abreviando el apellido del autor a las tres primeras letras, seguidas de los dos últimos dígitos del año de publicación, cerrando la cita entre corchetes: [Knu86]. En caso de tener dos o tres autores, se junta la primera letra del apellido de cada uno de los autores, y si son más de tres, se escriben las iniciales de los dos primeros seguidos de un signo «más» (+), por ejemplo: [PS95] y [GM+94].

Estilos bibliográficos

Para la presentación de la bibliografía, existen varios estilos, entre ellos se tiene: el estilo de notas, numérico, alfabético, alfabético abreviado y el estilo chicago.

- Con el estilo bibliográfico de notas, junto con el sistema numérico de citas, las referencias bibliográficas se disponen como notas a pie de página.
- El estilo bibliográfico numérico, usado juntamente con el sistema numérico de citas, compone las referencias, ya sea ordenadas alfabéticamente o por orden de citación, pero siempre al final del trabajo y llevan por etiqueta la cita bibliográfica correspondiente.

- El estilo bibliográfico alfabético, junto con el sistema autor-fecha abreviado de citas, dispone las referencias bibliográficas por orden alfabético al final del trabajo y su etiqueta correspondiente, en formato autor-fecha abreviado.
- El estilo bibliográfico alfabético abreviado es una variante del estilo bibliográfico alfabético. A diferencia del anterior, este estilo lleva el nombre de los autores abreviado con las iniciales.
- Por último, el estilo bibliográfico chicago, juntamente con el sistema autor-fecha, se compone el apellido antes del nombre del autor, y no se etiquetan.

1.2. Elaboración de documentos técnicos

La persona o el grupo de personas encargadas de elaborar un documento técnico deben seguir alguna metodología para lograr que el documento sea eficiente. En este trabajo se tomará la metodología descrita en González Videgaray (2004) y se muestran sus etapas en el diagrama en la figura 1.1.

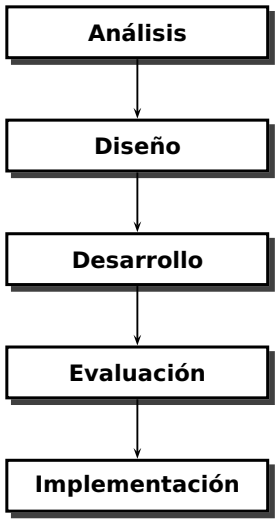


Figura 1.1: Etapas para crear un documento técnico

A continuación se explica cada una de estas etapas, sus actividades y productos resultantes.

1.2.1. Análisis

En esta primera etapa, el escritor o equipo de escritores deberán determinar los objetivos que se quieren lograr a través del documento. Tendrán que realizar entrevistas con la organización que lo solicite y hacer revisiones de documentos anteriores o relacionados con el tema. Una vez definidas las necesidades y los recursos disponibles se forma el equipo de trabajo y se determina la estrategia a seguir.

El producto que se obtiene del análisis es un plan de trabajo. Además, es necesario tener control sobre ese plan, es decir, el plan de trabajo debe cumplirse en los tiempos establecidos (por medio de

la elaboración de reportes de avance se puede tener este control). Los reportes de avance se tendrán que elaborar durante todo el proceso.

Actividades

- Definir los objetivos del documento y las necesidades a satisfacer
- Identificar los perfiles de los lectores potenciales
- Establecer una estrategia de documentación
- Formar el equipo de trabajo

1.2.2. Diseño

En la etapa de diseño se elabora la estructura principal del documento, también conocido como esquema general. Además, se crea la «guía de estilos». La guía de estilos se elabora considerando el tipo de personas que leerán el documento, ya que incluirá indicaciones del lenguaje, vocabulario, tipografía y colores que se usarán en todo el documento; todo el grupo de trabajo deberá seguir estos lineamientos.

Realizando lo anterior se obtiene como producto la estructura detallada del documento.

Actividades

- Definir las especificaciones de contenido
- Desarrollar la estructura del documento
- Diseñar la guía de estilos
- Diseñar documentos prototipo

1.2.3. Desarrollo

Aquí se procede a desarrollar el contenido del documento. Es indispensable seguir la guía de estilos establecida en la etapa anterior, ya que así se obtendrá un documento homogéneo y fácil de actualizar.

Los productos obtenidos en la etapa de desarrollo son documentos prototipo y la guía de estilos completa.

Actividades

- Crear documentos prototipo
- Obtener la aprobación de los prototipos
- Completar la guía de estilos
- Crear las ilustraciones
- Crear y revisar borradores

1.2.4. Evaluación

Antes de publicar el trabajo es importante realizar una evaluación. En esta etapa se revisa la forma, el contenido, la presentación, la ortografía y la redacción, entre otros aspectos más. Esta revisión se sugiere que la haga alguna persona ajena al grupo de trabajo, así, la evaluación es más confiable. Por último, en esta etapa, se crean las referencias del documento: apéndices, glosarios e índices.

El producto de esta etapa es el documento en su versión final, listo para su distribución.

Actividades

- Crear apéndices
- Crear glosarios
- Crear índices
- Revisar los documentos terminados
- Obtener la aprobación final

1.2.5. Implementación

Una vez terminado el documento y aprobado por la organización se procede a su distribución. Es importante, no obstante cómo se distribuya, que se mantenga la mejor calidad del documento. Asimismo, para facilitar las revisiones y adecuaciones necesarias del documento en cuestión, se tendrá que definir una estrategia para mantenerlo actualizado.

Los productos de esta etapa son: la documentación para crear los originales mecánicos, la producción final del documento y las memorias del proyecto.

Actividades

- Definir detalles de reproducción masiva
- Seleccionar materiales, empaque y medio de distribución
- Determinar las estrategias de mantenimiento y actualización
- Crear memorias del proyecto

1.3. Documentos técnicos utilizados en la carrera

Por último en este capítulo, se hablará acerca de los documentos técnicos que se utilizan dentro de la carrera de Matemáticas Aplicadas y Computación.

En la sección 1.1 se mencionaron algunos ejemplos de documentos técnicos, de los cuales la mayoría se utilizan dentro de la carrera, sin embargo, no todos son desarrollados por un estudiante de Matemáticas Aplicadas y Computación, cuando menos no frecuentemente; por esta razón sólo se considerarán los siguientes:

1. Documentación de software
2. Manuales de usuario

3. Presentaciones
4. Escritos científico-técnicos
    a) Trabajos de investigación
    b) Trabajos de titulación

1.3.1. Documentación de software

Durante todo el desarrollo de algún sistema, la documentación está presente. En cada etapa de la ingeniería de software se requiere de documentos para poder desarrollar el sistema de la mejor manera. En estos documentos se describe desde qué es lo que el cliente necesita hasta la planeación de la capacitación para los usuarios, además de que son útiles para futuras actualizaciones del sistema.

En este trabajo se considera como «documentación de software» al conjunto de documentos que se utilizan durante el desarrollo del mismo. Se menciona a continuación cómo se debe elaborar esta documentación. En esta sección no se consideran los manuales de usuario, ya que se tratarán en la sección siguiente.

Etapas de documentación

De manera general, el desarrollo de cualquier sistema computacional se divide en las siguientes etapas:

1. Análisis y definición de requerimientos
2. Diseño
    a) del sistema
    b) de programas
3. Implementación de programas
4. Pruebas
    a) unitaria
    b) de integración
    c) del sistema
5. Mantenimiento

El equipo encargado de desarrollar el sistema está formado por distintos tipos de personas, las cuales participan en etapas específicas y no siempre tienen contacto unas con otras. Por ello es necesario que se tenga documentada cada etapa.

Plan de proyecto Cuando se inicia cualquier proyecto de software es necesario definir un plan para especificar qué es lo que el cliente pide y con qué recursos se cuenta. Para esto se elabora un documento llamado «plan de proyecto».

El plan de proyecto se elabora para el equipo de desarrollo. En este documento se incluyen las siguientes partes:

Alcance del proyecto. En esta parte se delimita el sistema, explicando qué se incluirá y que no. Con esto se asegura que el equipo de desarrollo comprende lo que el cliente necesita.

Cronograma. En esta parte se describe cada una de las actividades a realizar y el tiempo que consumirá. Un diagrama de Gantt es útil para este propósito.

Organización del equipo de desarrollo. Aquí se define quiénes formarán parte del equipo de desarrollo y en qué parte del proceso colaborarán.

Descripción técnica del sistema propuesto. En esta parte se enlista el software y hardware que se ocupará para el sistema, se mencionan los compiladores, interfaces y algún equipo especial que se requiera. También se mencionan los métodos, herramientas y técnicas que se utilizarán a lo largo del proyecto.

Planes. Cuando un proyecto es grande, es necesario definir otros planes para considerar aspectos como: el aseguramiento de calidad, la configuración del sistema, la documentación, la gestión de datos y de recursos, las pruebas, el entrenamiento, la seguridad, la gestión de riesgo y el mantenimiento. En ocasiones, es mejor tener cada uno de estos planes en un documento por separado para que sea más fácil utilizarlos. Sin embargo, cuando el proyecto es pequeño, sólo se necesitan algunos de ellos. Estos planes se pueden consultar con más detalle en Pfleeger (2002).

Análisis y definición de requerimientos Un requerimiento es una característica del sistema o una descripción de algo que el sistema es capaz de hacer. Estos requerimientos los da el usuario cuando explica al desarrollador sus necesidades. Es importante que los requerimientos queden claros, tanto para el desarrollador como para el cliente, ya que de lo contrario se puede tener un sistema inservible para el éste.

Durante la determinación de los requerimientos, el desarrollador debe tener en cuenta lo siguiente:

Ambiente físico. En dónde se encuentra todo lo que el sistema necesita para funcionar, además de las restricciones ambientales, como temperatura, humedad o interferencia magnética, si existen.

Interfaces. Determinan las posibles entradas de información, así como las salidas del nuevo sistema. También se necesita saber si los datos requieren de algún formato especial.

Factores humanos. Quiénes utilizarán el sistema, si todos tienen las mismas habilidades, y determinar la clase de entrenamiento que se dará a cada tipo de usuario.

Funcionalidad. Saber qué hará el nuevo sistema y cuándo lo hará, además de saber cómo y cuándo se puede cambiar o modificar el sistema.

Documentación. Qué tipo de documentación se requerirá para los usuarios y cómo se distribuirá (esta parte se trata más a detalle en la sección 1.3.2).

Datos. Cuál será el formato de los datos, tanto los de entrada como los de salida, así como la precisión que requerida en los cálculos que el sistema realizará.

Recursos. Conocer con qué recursos se cuenta, tanto económicos, físicos (materiales y espacio disponible) y de personal.

Seguridad. Conocer si se deberá controlar el acceso al sistema o a la información y establecer cómo se controlará esto. Además, se debe saber con qué frecuencia se harán las copias de seguridad y si se necesita tomar otras precauciones (contra fuego, robo o algún otro siniestro).

Aseguramiento de la calidad. Por último, se deben tomar consideraciones con respecto a la calidad del sistema, por ejemplo: cómo determinar su confiabilidad, si existe un tiempo máximo permitido para su recuperación después de una falla, cómo se podrán hacer cambios en el diseño después de implementado, si se transportará frecuentemente de una computadora a otra y si el mantenimiento incluirá hacer actualizaciones o sólo se corregirán errores.

Tomando los puntos anteriores se elaborarán dos documentos, uno dirigido al cliente y otro dirigido al equipo de desarrollo.

- El documento denominado «definición de requerimientos» es el dirigido al cliente. Debe estar escrito de forma que el cliente pueda comprender fácilmente lo que hará el sistema.
- El segundo documento, la «especificación de requerimientos» se dirige al equipo de desarrollo.

En ocasiones, los requerimientos se pueden resumir en un solo documento.

El documento de la especificación de requerimientos es más extenso que la definición de los mismos, ya que se explica detalladamente el funcionamiento del sistema propuesto. Aquí se recomienda comenzar con una descripción general del sistema y luego ir desglosando cada módulo.

La forma de describir los requerimientos puede ser por medio de fórmulas matemáticas, definiciones axiomáticas, tablas de decisiones, tablas de eventos, diagramas de transición, diagramas UML, diagramas de flujo de datos, etc. Sea cual sea la forma de describir los requerimientos, debe ser comprensible para todo el equipo de desarrollo.

Diseño del sistema Una vez aceptados los requerimientos, la siguiente etapa es diseñar el sistema. Durante esta etapa se crea, de cada requerimiento (problema), una solución. Generalmente se elaboran dos documentos: el primero, «diseño conceptual», explica al cliente el «qué» hará el sistema para dar solución a su problema; el segundo, «diseño técnico», está dirigido al equipo de desarrollo, aquí se explica el «cómo» el sistema funcionará, detallando cada uno de sus módulos. Cuando el cliente acepta el diseño conceptual se elabora el diseño técnico.

Ambos documentos incluyen una sección denominada justificación racional del diseño, donde se explica cómo y por qué se hará así el sistema.

Estos documentos también incluyen una descripción de los componentes del sistema; entre ellos se debe mencionar cómo el usuario interactúa con él, considerando lo siguiente:

- Los menús y las interfaces del sistema, mencionando las teclas de función, uso del mouse o joystick y otros periféricos especiales
- Los formatos para los informes
- La entrada y salida de datos, el formato que tendrán y dónde se almacenarán
- La topología de la red y restricciones de seguridad

Al igual que en la especificación de requerimientos, es común utilizar diagramas para explicar mejor todo lo anterior.

Por último, estos documentos deben mencionar cómo es que se cumplirá con cada uno de los requerimientos.

Diseño de los programas Una vez que se ha completado el diseño del sistema y que el cliente lo ha aceptado, es momento de hacer los programas.

Cada código fuente debe ser documentado, de tal manera que cualquier otra persona del equipo sepa quién escribió el programa, además de que pueda entender qué hace cada módulo escrito. A esta documentación se le conoce como «documentación interna» de los programas.

Existe también la llamada «documentación externa» de los programas, la cual es más útil que la anterior, ya que pocas personas tendrán acceso al código fuente del sistema (pero no implica que deba omitirse la documentación interna). Además, en la documentación externa se explica con más detalle cada módulo programado.

La documentación externa debe describir claramente cómo funciona cada módulo del sistema, los algoritmos que se utilizan, los tipos de datos empleados y el flujo de los mismos. También se debe dar una justificación de por qué se eligió ese método en particular para solucionar el problema.

Prueba del sistema Una vez que se ha terminado con la programación del sistema, es necesario revisar que no existan fallas; para esto se realizan varias pruebas sobre él. La etapa de pruebas se divide en otras más, comenzando con lo más simple del sistema y progresivamente se incorporan todos los elementos que intervendrán en el mismo.

Las distintas etapas dentro de las pruebas son:

Pruebas unitarias. Se inicia probando cada módulo, verificando que realice lo especificado en el diseño.

Prueba de integración. Una vez que todos los módulos funcionan correctamente, se integran y se revisa que funcionen según el diseño.

Prueba de función. Aquí se evalúa el sistema comparándolo con los requerimientos, para ver si realiza lo que el cliente solicitó.

Prueba de rendimiento. En esta etapa se revisa qué tan confiables son los los resultados generados por el sistema, además del tiempo de respuesta. Si existen algunas restricciones de hardware o software, el sistema se prueba con dichas restricciones.

Prueba de aceptación. Cuando el sistema ha pasado las pruebas anteriores, éste se muestra al cliente para ver si cumple con sus espectativas.

Prueba de instalación. Por último, el sistema se instala en el lugar donde será utilizado y se revisa que todo funcione correctamente.

Cada una de estas pruebas debe ser documentada, con el fin de poder corregir los defectos encontrados. Son varios los documentos producidos en esta etapa.

El primer documento es el «plan de prueba». En el plan de prueba se describen de manera general cada una de las pruebas que se le aplicarán al sistema. Para cada prueba se debe incluir lo siguiente:

Objetivos. Se da el propósito de la prueba, explicando cómo es que evaluará al sistema. Se mencionan los requisitos para poder realizar la prueba. También se explican los resultados esperados por ella.

Referencias a documentos. En esta sección se hace referencia a otros documentos, como los requerimientos y el diseño del sistema, para definir la o las pruebas que se realizarán.

Casos de prueba. Se definen los casos de prueba, es decir, qué aspectos del sistema se evaluarán, con el fin de que las pruebas sean útiles para su revisión.

Cronogramas. Aquí se define el lugar donde se realizará la prueba, así como su programa. Se incluyen los tiempos de inicio y término y de cualquier requerimiento especial (por ejemplo, la generación de datos), también se incluye el tiempo necesario para preparar y revisar los informes.

Materiales necesarios. Por último, si se necesitan algunos materiales para realizar la prueba, se especifican en esta sección.

Una vez definido el plan de prueba, se elabora por cada prueba su «especificación». Aquí se enlistan los requerimientos que se verificarán y se explica su propósito. También se explican las condiciones sobre las que se realizará la prueba y los métodos que se utilizarán para evaluarla.

Otra parte que se incluye en esta documentación es la «descripción de la prueba». Este documento explica paso a paso cómo se realizará la prueba. La descripción debe incluir lo siguiente:

Medios de control. Si la prueba será iniciada y controlada por medios automáticos o manuales.

Datos. Se explican los datos de entrada, comandos de entrada, estados de entrada, datos de salida, estados de salida y mensajes producidos por el sistema durante la prueba. Se explica también cómo iniciar la prueba, detenerla o suspenderla, o bien repetir o retomar una prueba incompleta, o terminar la prueba.

Procedimiento. También conocido como «guión de prueba», proporciona una descripción paso a paso de cómo realizar la prueba.

Por último, es necesario analizar los resultados obtenidos por cada una de las pruebas para saber si se cumplen los requerimientos. A esta parte se le conoce como «informe de análisis de prueba». Aquí, además de conocer si el sistema funciona correctamente, se miden otros aspectos, por ejemplo, la velocidad de cálculo y la exactitud de los resultados. Esto permite conocer si el sistema está completo, además de conocer su confiabilidad.

Mantenimiento Después de entregarle al cliente el sistema funcionando, es importante considerar el mantenimiento del mismo. El mantenimiento de un sistema computacional se clasifica en dos: el que repara defectos y el que mejora la aplicación.

Para mantener el sistema, se deben seguir las mismas etapas que se mencionaron al inicio de esta sección (página 15), pero enfocándose en el defecto a corregir o la mejora del sistema. Estas etapas son las siguientes:

Identificación Ya sea el cliente o el equipo de desarrollo detectan el error o la posible mejora. Cuando se detecta un error, el equipo de desarrollo es el responsable de registrar todo el procedimiento que lo generó, para poder reproducirlo posteriormente.

En ambos casos, los puntos a documentar son: la entrada (qué es lo que genera el error o qué se quiere mejorar), el proceso y la salida (lo que se espera que el sistema haga), además de que se deben mantener los estándares de calidad en todo momento.

Análisis, diseño, implementación y pruebas El procedimiento para el resto de las etapas es el mismo que se sigue en el desarrollo de un sistema completo, pero enfocándose a lo establecido en la etapa anterior.

Estructura general

El proceso de desarrollo de software incluye varios documentos, ya que sin ellos el sistema producido puede no ser lo que el cliente espera. En cada una de estas etapas se requiere de cuando menos un documento, y en ocasiones se requieren dos versiones: una para el cliente y otra para el equipo de desarrollo; además, muchos de estos se desarrollan en conjunto con el sistema.

Todos los documentos mencionados anteriormente se resumen en el cuadro 1.1.

Etapa de desarrollo	Documento dirigido a	
	equipo de desarrollo	cliente
Planeación	Plan de proyecto	—
Requerimientos	Especificación de requerimientos	Definición de requerimientos
Diseño	Diseño técnico	Diseño conceptual
Pruebas	Plan de pruebas	—
Mantenimiento	Mantenimiento del sistema	—

Cuadro 1.1: Documentos generados durante el desarrollo del software

Para definir una estructura para todos estos tipos de documentos, se toma la estructura mencionada en la sección 1.1.2. Entonces, la estructura para la documentación de software queda como se observa en la figura 1.2.

Preámbulo

- Portada
- Resumen
- Tabla de contenido

Contenido

- Desarrollo

Referencias

- Apéndices
- Glosarios
- Índices

Figura 1.2: Estructura para la documentación de software

Sus elementos se explican a continuación.

- En la portada se incluye:
  - Nombre y versión del sistema
  - Tipo de documento, por ejemplo: «Especificación de requerimientos», «Diseño del sistema», etc.

  - Nombre de la persona o personas quienes elaboraron el documento
  - Fecha

- El resumen es útil para el equipo, ya que describe rápidamente el contenido del documento. No es necesario cuando el documento va dirigido al cliente.
- Los apéndices son opcionales. En estos se puede incluir alguna información complementaria al documento, por ejemplo, algunos conceptos teóricos que manejará el sistema o los resultados de las pruebas.
- Los glosarios son útiles para que todo el equipo de desarrollo entienda los conceptos que se estén manejando. En ocasiones se colocan en el preámbulo, después del resumen.
- Por último, los índices ayudan a localizar los conceptos más rápido. Sin embargo, son opcionales, ya que los documentos no son demasiado largos.

El desarrollo dependerá del tipo de documento que se esté realizando. Para los documentos del equipo de desarrollo, existen varios estándares para la documentación de software. Para este trabajo se toman como base los estándares mencionados en Braude (2003) y se ajustan a los puntos explicados anteriormente. En los cuadros 1.3, 1.4, 1.5, 1.6 y 1.7 se muestran los contenidos para cada documento.

Los documentos dirigidos al cliente tratan de manera general los mismos puntos, pero omitiendo detalles técnicos y son escritos en un lenguaje comprensible para el lector.

1. Introducción

1.1. Alcance del proyecto
1.2. Entregas del proyecto

2. Organización del equipo de desarrollo

2.1. Asignación de personal
2.2. Mecanismos de supervisión y control

3. Descripción técnica

3.1. Métodos, herramientas y técnicas
3.2. Plan de documentación
3.3. Requerimientos de recursos
3.4. Otros planes (opcional)

4. Cronograma

Figura 1.3: Contenido del plan de proyecto

1. Descripción general

1.1. Ambiente físico
1.2. Factores humanos
1.3. Documentación
1.4. Flujo de datos
1.5. Recursos disponibles
1.6. Restricciones de seguridad
1.7. Aseguramiento de la calidad

2. Requerimientos específicos

2.1. Requerimientos funcionales
2.2. Requerimientos no funcionales

Figura 1.4: Contenido de la especificación de requerimientos

1. **Descripción de descomposición**
   - 1.1. Descomposición de módulos
   - 1.2. Descomposición de procesos
   - 1.3. Descomposición de datos
2. **Descripción de dependencia**
   - 2.1. Dependencias de módulos
   - 2.2. Dependencias de procesos
   - 2.3. Dependencias de datos
3. **Descripción de interface**
   - 3.1. Interface de módulo
   - 3.2. Interface de proceso
4. **Diseño detallado**
   - 4.1. Diseño detallado de módulos
   - 4.2. Diseño detallado de datos

Figura 1.5: Contenido del diseño técnico

1. **Plan de pruebas**
   - 1.1. Objetivos
   - 1.2. Cronogramas
2. **Especificación de la prueba**
   - 2.1. Referencias a documentos
   - 2.2. Casos de prueba
   - 2.3. Materiales necesarios
3. **Descripción de la prueba**
   - 3.1. Medios de control
   - 3.2. Datos
   - 3.3. Procedimientos
4. **Informe de análisis de prueba**

Figura 1.6: Contenido de la prueba

1. **Identificación del problema o mejora**
   - 1.1. Entrada
   - 1.2. Proceso
   - 1.3. Control
   - 1.4. Salida
2. **Análisis**[a]
3. **Diseño**
4. **Implementación**
5. **Pruebas del sistema**
6. **Pruebas de aceptación**

[a] En ésta y las demás secciones se consideran los mismos puntos de la sección 1.

Figura 1.7: Contenido del mantenimiento

1.3.2. Manuales de usuario

Una vez terminado el sistema es importante que el usuario sepa cómo operarlo, ya que si no se sabe utilizar, el sistema no tendrá éxito. Existen varias formas de capacitar a los usuarios para que aprendan a utilizar el sistema, entre ellas se encuentran:

Documentos. Estos documentos contienen toda la información necesaria para utilizar el sistema de manera correcta y eficiente. Estos documentos deben ser accesibles para los usuarios. Un manual de este tipo debe estar organizado de tal manera que el usuario pueda encontrar fácilmente la información que busca, ya que comúnmente, un manual sólo se consulta cuando el sistema falla.

Iconos y ayuda en línea. Actualmente, las interfaces de los sistemas computaciones están acompañadas de iconos que facilitan la comprensión de sus funciones. La ventaja de utilizar iconos es que resulta más fácil recordar imágenes que comandos o sintaxis.

De igual manera, muchos sistemas cuentan con ayuda en línea que facilita la capacitación. Es más fácil buscar información directamente en documentos electrónicos que buscar en un documento impreso.

Demostraciones y cursos. Las demostraciones y cursos son muy utilizadas para capacitar a los usuarios, ya que pueden diseñarse para cubrir aspectos específicos del funcionamiento del sistema, además de que estos resultan más dinámicos que un documento impreso o electrónico.

Este tipo de capacitación también permite reforzar lo aprendido. Escuchar, leer y observar cómo trabaja una función ayuda a recordar las funciones más fácilmente.

Sea cual sea la forma de capacitar, siempre se necesitará elaborar un manual para el usuario, ya sea impreso o en formato electrónico.

Creación del manual de usuario

Como se mencionó en la metodología para crear documentos técnicos, en la etapa de análisis para la creación del manual de usuario se debe considerar el tipo de personas que utilizarán el manual. No siempre tienen las mismas necesidades los usuarios potenciales del sistema, así que pueden crearse distintos manuales, cada uno de ellos con un propósito distinto. Un ejemplo de esto se puede ver en Pfleeger (2002), aquí se mencionan los distintos manuales del programa S-PLUS, estos son:

- *A Gentle Introduction to* S-PLUS. Un manual de nivel elemental para un usuario de computadora novato.
- *A Crash Course in* S-PLUS. Para usuarios con conocimientos de computación avanzados.
- S-PLUS *User's Manual.* Este manual explica cómo comenzar a utilizar el programa, manipular datos y utilizar gráficos avanzados.
- S-PLUS *Guide to Statistical and Mathematical Analysis.* Aquí se describe el modelado estadístico.
- S-PLUS *Programmers Manual.* Este manual explica los lenguajes de programación S y S-PLUS.
- S-PLUS *Programmers Manual Supplement.* Un suplemento con información específica para la versión del software.

- S-PLUS *Trellis Graphics User's Manual.* Describe las características gráficas particulares para complementar el análisis estadístico.
- S-PLUS *Global Index.* Un índice que proporciona una referencia cruzada entre los distintos manuales.

Como se puede ver, cada manual está dirigido a un grupo de usuarios específico, que varían en sus necesidades. Sin embargo, el formato general de los manuales es el mismo.

Estructura general

Un manual de usuario sirve como una guía de referencia o tutorial completo y comprensible para los usuarios del sistema. En ocasiones de presenta el sistema por capas, es decir, se comienza con un propósito general y se avanza hasta dar una descripción detallada de cada función.

En la primera parte del documento, se describe el propósito del manual, así como referencias a otros documentos que contienen información más detallada. Esto es útil para que el usuario conozca si encontrará lo que busca en el manual. También se incluye una lista con los términos especiales, abreviaturas y acrónimos que se utilizarán en el manual.

Un manual, para que pueda ser funcional, debe contemplar los siguientes puntos:

- Mapa de los componentes principales y su relación entre sí
- Descripción de las distintas pantallas que pueden aparecer en la ejecución del sistema y el propósito de cada una
- Descripción de cada opción del menú y de las teclas de función
- Descripción de todas las entradas esperadas por cada función
- Descripción de todas las salidas posibles por función
- Descripción de las características especiales que posee cada función

A lo largo de este documento es útil incluir ilustraciones para complementar el texto, ya que ayudan a que el usuario comprenda mejor el funcionamiento del sistema. De igual manera, incluir información en distintas tablas en ocasiones es mejor que escribir párrafos que pueden aburrir al lector.

En la figura 1.8 se muestra la estructura de un manual de usuario y enseguida se da una explicación de sus secciones.

- La portada incluye lo siguiente:
  - Título del manual
  - Nombre y versión del sistema (si no se incluye en el título del manual)
  - Nombre del autor o autores
- En la nota de la edición se incluyen datos acerca del libro, como son: fecha de publicación, número de edición, copyright y permisos de las marcas registradas utilizadas a lo largo del libro. Comúnmente esta información se coloca en al reverso de la portada.

Preámbulo

- Portada
- Nota de la edición
- Licencias
- Prefacio
- Tabla de contenido

Contenido

- Desarrollo

Referencias

- Apéndices
- Glosarios
- Índices

Figura 1.8: Estructura para un manual de usuario

- La parte de licencias explica al usuario los términos y condiciones de uso del sistema y las garantías y limitaciones del usuario. En ocasiones se incluye algún domicilio, teléfono, sitio web o correo electrónico para atención al usuario.
- El prefacio es una introducción al sistema, se describe de manera rápida las actualizaciones, se delimita la función del manual y se explica la manera en que éste se organiza.
- En la parte del contenido, se desarrolla el tutorial o guía de uso del sistema, según el propósito del manual.
- Los apéndices contienen información que complementan el manual. No debe incluirse en un apéndice información que sea fundamental para comprender el funcionamiento del sistema.
- Los glosarios dan una explicación del significado de palabras técnicas que pueden causar duda al lector.
- Por último, los índices se elaboran para ayudar al usuario a localizar rápidamente la información que necesita. Los índices deben dar mayor información que la tabla de contenido.

1.3.3. Presentaciones

Las presentaciones orales son otro tipo de documento técnico que se utiliza con frecuencia. Después de elaborar algún trabajo de investigación, es común que se tenga que presentar ante un grupo de personas: los compañeros de clase, los sinodales, los directivos de una empresa, etc.

Elaboración de presentaciones

Las presentaciones solían hacerse con acetatos o diapositivas, en estos se colocaba la información y los gráficos y con ayuda de un proyector se presentaban al grupo. Actualmente se cuenta con gran

variedad de programas de cómputo que ayudan a hacer presentaciones para, ya sea imprimirse en acetatos, o para proyectarse directamente de la computadora con ayuda de un cañón.

Durante la elaboración de una presentación, se debe tener en cuenta el propósito de la misma, es decir, qué es lo que se pretende dar a conocer. Una presentación puede servir para: 1) informar, 2) instruir, o 3) persuadir.

Propósito informativo. Una presentación puede servir para dar información al grupo, por ejemplo, acerca de los avances de cierto proyecto de trabajo, o de las actualizaciones en el equipo de trabajo.

Propósito instructivo. Este tipo de presentaciones se suelen utilizar en las escuelas, cuando el profesor enseña algún tema. También pueden utilizarse para capacitar a los empleados de una compañía acerca del equipo de trabajo adquirido recientemente.

Propósito persuasivo. El propósito de estas presentaciones es convencer o aconsejar al grupo para tomar alguna decisión. Por ejemplo, cuando se propone algún proyecto ante los directivos, lo que se pretende con la presentación es convencer al grupo que el proyecto es bueno y dará ciertos beneficios a la empresa. De igual manera, cuando se vende un producto lo que se pretende es que los clientes se convenzan de que les será útil.

Sea cual sea el propósito de la presentación, se deben tomar en cuenta algunos puntos para que sea exitosa:

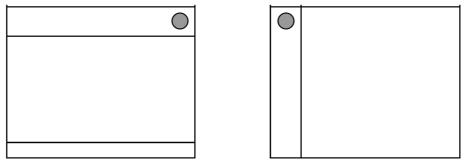
- El tiempo de la presentación no debe exceder de los 7 minutos. Presentaciones más largas hacen se pierda el interés.
- La persona que estará dando la presentación debe cuidar su tono de voz, sus gestos y ademanes.
- La presentación debe tener un objetivo, es decir, qué se pretende dar a conocer. El objetivo puede variar de acuerdo al tipo de gente que estará escuchando la presentación.
- Se debe tener bien organizado el tema, y considerar los elementos visuales que se incluirán (gráficos, tablas y animaciones).
- La presentación siempre debe tener una conclusión. Si no se concluye correctamente, la presentación puede no tener el efecto deseado.

Asimismo, el diseño de una presentación es importante para que se logre el objetivo. Los elementos que se deben tener en cuenta son los siguientes:

- En cada diapositiva o acetato se deben incluir ideas completas y cortas, ya que sólo es una guía para el expositor.
- La tipografía dentro de las diapositivas deberá ser sans serif, de tamaño grande, para que pueda ser legible a lo lejos. Los títulos deben escribirse con tipografía más grande, según su jerarquía.
- No es recomendable justificar a ambos márgenes el texto de la tipografía, es preferible justificar al margen izquierdo.
- La diapositiva se puede dividir en dos o tres partes:
  - Primera: para el título o tema, o la imagen corporativa.

- Segunda: para el texto principal.
- Tercera (opcional): para la fecha o datos complementarios.

Por ejemplo:



- Si la presentación se hace a color, deben utilizarse colores contrastantes y que no lastimen al público. Puede utilizarse un color para resaltar ideas, otro para los títulos y otro para el texto general. El fondo debe ser el mismo en toda la presentación, a menos que se justifique un cambio.
- Las ilustraciones y gráficos deben ser claros en su objetivo y presentación. Si la presentación no se hace a color, es preferible utilizar tramas en los gráficos, en lugar de tonos de gris.

Estructura general

Sin importar el propósito de la presentación, ésta tiene una estructura básica para que sea funcional. Dicha estructura se muestra en la figura 1.9 (también se da una explicación de cada una de sus secciones).

Preámbulo

- Portada

Contenido

- Introducción
- Desarrollo
- Conclusiones

Referencias

- Fuentes

Figura 1.9: Estructura para una presentación

- En la portada se incluye el nombre de la institución o compañía, el título de la presentación y el nombre del expositor, junto con su correo electrónico o algún otro medio donde se le pueda localizar. La portada puede tener un diseño distinto que el resto de las diapositivas, pero debe conservarse la imagen.
- En la parte del contenido se explica todo el tema, tomando los puntos que se muestran en la figura:

  - Primero se da una introducción al tema, mencionando el objetivo de la presentación.
  - En la parte de desarrollo se explica el tema, sus conceptos fundamentales, métodos utilizados para el trabajo y los resultados obtenidos.
  - Por último, se deben dar las conclusiones del trabajo realizado, o lo que se pretende lograr si se expone algún proyecto.

- En la parte de referencias se presentan las principales fuentes de información sobre las cuales se desarrolló el trabajo

1.3.4. Escritos científico-técnicos

A lo largo de la carrera, es común que los profesores requieran un trabajo de investigación acerca de algún tema específico, por ejemplo: aplicaciones de las series de Fourier en la medicina. Este tipo de trabajos se les conoce como «escritos científicos».

Un escrito científico entra en la categoría de documentos técnicos porque contiene información acerca de un área del conocimiento y se presenta de manera estructurada. Además, este tipo de escritos van dirigidos a un grupo específico de lectores, como son los profesores y alumnos de la carrera. Por estas razones se conocen también como «escritos científico-técnicos».

Este tipo de trabajos escritos frecuentemente se piden como proyectos de fin de curso y son conocidos también como «trabajos de investigación». Además de estos se tienen los «trabajos de titulación», que son requisito para obtener el título, por ejemplo: tesis, tesina, memorias de desempeño profesional, reporte de servicio social, etc. Sin embargo, ambos tipos de escritos tienen como objetivo que el estudiante aplique los conocimientos de una disciplina de una manera original, útil y correcta.

Elaboración de un escrito científico-técnico

Para elaborar un escrito científico se toma casi íntegra la metodología descrita anteriormente (sección 1.2). Las diferencias se explican a continuación.

Análisis Durante esta primera etapa se plantea un objetivo, es decir, qué problema se va a tratar; en los trabajos de titulación se le conoce como hipótesis. Una vez definido el objetivo se puede elaborar un esquema para saber qué temas se tomarán como base de la investigación, y posteriormente incluirlos en el trabajo.

Otro punto a considerar es quiénes serán los lectores potenciales del trabajo, comúnmente los profesores y estudiantes de la carrera o carreras afines. Es importante tener esto presente, ya que evitará que se tengan trabajos demasiado extensos.

Diseño En esta etapa se fija el contenido del trabajo, partiendo el esquema hecho en la etapa anterior. Además se fijan las normas de estilo que utilizará el trabajo. Estas normas de estilo comúnmente las fija la escuela, y en tal caso se deben seguir al pie de la letra; en caso contrario pueden tomarse de alguna otra institución.

Desarrollo Una vez que se define la estructura o esqueleto del trabajo, se procede a la investigación y documentación del mismo. Durante la documentación de la investigación se elabora el borrador

del trabajo y se recopila la bibliografía utilizada. Se tienen que elaborar también las ilustraciones y tablas que necesite el trabajo.

Evaluación Esta etapa consiste en revisar el borrador y se corrige o cambia lo necesario. La persona que revisa el borrador puede ser el mismo profesor que dejó el trabajo y en el caso de los trabajos de titulación el asesor es quien lo hace.

Además, en esta etapa es cuando se crean los apéndices que se necesiten. Los apéndices no deben contener información relevante para la comprensión del trabajo, más bien son complementarios al mismo.

Esta etapa de evaluación se puede repetir tantas veces como sea necesario para obtener un buen trabajo.

Implementación Aquí simplemente se entrega el trabajo. En el caso de los trabajos de titulación, consiste en la reproducción del documento para los sinodales y la escuela.

Estructura general

Los dos tipos de trabajo que se han mencionado tienen la misma estructura en la parte del contenido y referencias, es en el preámbulo donde se tienen pequeñas variaciones.

Trabajo de investigación El esquema de un trabajo de investigación se muestra en la figura 1.10. De igual forma se explican sus secciones.

- La portada incluye los siguientes datos:
  - Nombre de la escuela
  - Título completo del trabajo
  - Nombre de la materia
  - Nombre del autor
  - Fecha
- El resumen, también conocido como «abstract», es una versión precisa y abreviada del trabajo, de no más de 200 palabras, y debe dar la información suficiente para que los lectores determinen si es útil o no leer todo el trabajo, además de que incluyen los principales objetivos del trabajo, los métodos empleados, los resultados y conclusiones más importantes.

  En algunas ocasiones es necesario incluir el resumen en el idioma del trabajo y en inglés.
- Las palabras clave, conocidas como «keywords», son una lista de cuatro a ocho términos descriptivos del contenido del documento. Esta lista permite que el documento sea clasificado dentro de las bases de datos de las bibliotecas.

  Al igual que el resumen, las palabras clave suelen escribirse tanto en el idioma del trabajo como en inglés.
- En la parte del contenido se tiene en primer lugar la introducción. Esta sección da una descripción del trabajo, además de que el autor explica los motivos del estudio y qué se pretende lograr con él. También se incluye un resumen de cada capítulo.

Preámbulo

- Portada
- Resumen
- Palabras clave
- Tabla de contenido
- Índice de figuras o ilustraciones
- Índice de cuadros o tablas

Contenido

- Introducción
- Desarrollo
- Resultados
- Conclusiones

Referencias

- Apéndices
- Fuentes
- Notas

Figura 1.10: Estructura para un trabajo de investigación

- El desarrollo del trabajo se puede dividir en varias secciones. Se comienza con un «marco de referencia» donde se delimita la investigación. Después se tiene un «marco teórico», aquí se dan las bases teóricas sobre las que se fundamenta el trabajo. Luego sigue el «planteamiento», donde se explica qué se hará, por qué de esa manera y la metodología.
- En la sección de resultados, se explica lo que se obtuvo al ejecutar lo expuesto en la sección anterior.
- Por último se tienen las conclusiones. Aquí se explica si se cumplió el objetivo o no.
- En la parte de anexos se tienen los apéndices. De igual forma que en los manuales de usuario, contienen información que complementa al documento, por ejemplo, los resultados de una encuesta, gráficas, etc. Los apéndices no deben ser muy extensos (no más que el contenido).
- Todas las fuentes de información consultadas deberán ir en la sección de fuentes. Para presentar esta parte de manera adecuada se siguen ciertos formatos o estilos (puede consultarse Turabian (1987) para mayor información).
- En la sección de notas se colocan todas las notas a pie de página del documento. Esta sección es opcional, ya que comúnmente las notas a pie de página se colocan en la misma página donde se originan.

Trabajo de titulación El esquema de un trabajo de titulación se muestra en la figura 1.11.

Preámbulo

- Portada
- Resumen
- Palabras clave
- Dedicatoria
- Agradecimientos
- Epígrafe
- Tabla de contenido
- Índice de figuras o ilustraciones
- Índice de cuadros o tablas

Contenido

- Introducción
- Desarrollo
- Resultados
- Conclusiones

Referencias

- Apéndices
- Fuentes
- Notas

Figura 1.11: Estructura para un trabajo de titulación

Como se mencionó, la estructura es casi igual a los trabajos de investigación, salvo por tres secciones que se agregan: dedicatoria, agradecimientos y epígrafe. Estas secciones no son obligatorias, pero con frecuencia se utilizan en este tipo de trabajos. El contenido de estas secciones se explica a continuación:

- La dedicatoria es un texto corto que menciona a quién se le dedica el trabajo.
- Los agradecimientos se utilizan para reconocer a las personas o instituciones que ayudaron de manera directa o indirecta a la elaboración del documento.
- El epígrafe es una cita de algún otro texto junto con el nombre del autor. En ocasiones se coloca al principio del documento o al comienzo de cada capítulo. Esta parte tiene el mismo formato que una dedicatoria.

Por último, en la portada se cambia el nombre de la materia por el grado que se obtendrá.

Capítulo 2

El lenguaje LaTeX

En el capítulo anterior se han dado los conceptos fundamentales acerca de la documentación técnica: qué es, para qué se utiliza y cómo se elaboran documentos técnicos; además de que se explicó con detalle cuáles son los documentos técnicos más utilizados por los estudiantes de Matemáticas Aplicadas y Computación.

Actualmente las computadoras ayudan a realizar muchas tareas que antes eran más difíciles de hacer, y también es el caso de la documentación técnica. Existen varios programas que sirven para elaborar documentación técnica, entre ellos se tiene: FRAMEMAKER, VENTURA PUBLISHER, FREEHAND, MS-WORD, LaTeX, entre otros; y la persona que desee hacer documentos técnicos necesita conocer alguno de estos.

En este trabajo se utilizará LaTeX para elaborar documentos técnicos, y en este capítulo se dará una breve descripción de él.

2.1. Orígenes

LaTeX tiene como antecesor a TeX. TeX es un lenguaje de programación creado por Donald Ervin Knuth entre los años 1977 y 1978, quien dice al comienzo de su libro *The TeXbook*, Knuth (1986):

> Amable lector, esto es un manual sobre TeX, un nuevo sistema de tipografía cuya intención es crear libros bellos, especialmente aquellos que contienen muchas matemáticas. Cuando prepare un manuscrito en formato TeX, le estará diciendo al ordenador cómo exactamente tiene que ser transformado en páginas cuya calidad tipográfica es comparable a la de las mejores imprentas del mundo...

Como se observa, no es lenguaje de programación común, como JAVA o C++, sino que se enfoca a la creación de documentos.

El nombre TeX tiene su origen en la raíz griega para referirse a tecnología: `teq`, que también hace referencia al arte, ya que Knuth expresa también en su libro: «con TeX, la meta es producir la máxima y más bella calidad [en un documento]».

Algunos años después, TeX fue presentado por Knuth a la AMS (*American Mathematical Society*), donde más tarde se desarrolló una versión de TeX para documentos científicos: $\mathcal{AMS}$-TeX. TeX pronto se difundió por las Universidades y actualmente se utiliza también en editoriales reconocidas a nivel mundial.

En 1982 Leslie Lamport creó LaTeX, una colección de comandos que simplifican el mecanografiado de un documento en TeX. El objetivo de Lamport era que el escritor se concentrara en la estructura del texto, en lugar de los comandos para darle formato. En el prólogo de *LaTeX — A Document Preparation System*, Lamport (1994), Lamport comentaba:

> Al transformar TeX en LaTeX, he tratado de convertir un coche de carreras muy bien afinado en un confortable sedán familiar. El sedán familiar no está pensado para ir tan rápido o ser tan excitante de conducir como el coche de carreras, pero es confortable y le llevará al supermercado sin alborotos. Sin embargo, LaTeX tiene toda la potencia de TeX escondida bajo el capó, y el conductor más aventurado puede hacer con él todo lo que puede hacer con TeX.

Posteriormente, LaTeX se ha ido actualizando y, al igual que TeX, es muy utilizado en Universidades y en editoriales reconocidas a nivel mundial. Sin embargo, TeX no ha sido desplazado por LaTeX, ya que algunos prefieren TeX para tener el control total de la composición del documento.

2.2. Características

Como cualquier programa, LaTeX posee ciertas características, algunas buenas y algunas malas, pero depende de con quien se compare. Lo más usual es compararlo con sistemas del tipo WYSIWYG, como MS-WORD; en estos sistemas el documento final será tal cual se muestra en pantalla (WYSIWYG, *What You See Is What You Get*, lo que ve es lo que obtendrá). Las principales características de LaTeX son las siguientes:

- Facilita la composición de fórmulas.
- Sólo se tienen que escribir instrucciones sencillas para indicar la estructura del documento; de manera similar a HTML.
- Las estructuras más complejas, como notas a pie de página, bibliografía, índices, tablas y otras, pueden producirse sin mucho esfuerzo.
- Existen paquetes y programas adicionales para muchas tareas que no se incluyen directamente en LaTeX.
- Funciona en varias plataformas y es gratis.
- Obliga a quien lo utiliza a estructurar su trabajo antes de capturar.
- Se requiere de bastante memoria al momento de crear un documento.
- La creación de un diseño entero no es tan fácil.

Algunas de estas características merecen una más amplia explicación.

En primer lugar, el punto más fuerte de (LA)TeX[1] es la composición de fórmulas matemáticas. En programas como MS-WORD existe un complemento que se llama *Editor de ecuaciones*, una versión de MATHTYPE, el cual ayuda a la creación de ecuaciones, pero es interesante saber que su núcleo de composición es TeX; en otras palabras, cuando se utiliza el *Editor de ecuaciones* se está utilizando una aplicación visual de (LA)TeX, exclusiva para fórmulas. De igual manera, gran variedad de programas

[1] Este logotipo se refiere tanto a TeX como a LaTeX.

matemáticos, como MAPLE, MATHEMATICA y MATLAB utilizan (LA)TEX para presentar las fórmulas en pantalla y es posible exportar los archivos a formato (LA)TEX.

Acerca de la sintaxis de LATEX, es semejante a HTML. Por ejemplo, dentro de LATEX, para indicar el título de un capítulo se escribe `\chapter{`*Nombre del capítulo*`}`, en lugar de escribir el *Nombre del capítulo* con una tipografía en negrita de 18 puntos. De manera similar funciona para crear otros elementos como referencias cruzadas, tablas y bibliografías. Esto se verá más adelante.

En cuanto a los paquetes, es similar a los lenguajes de programación más conocidos. Por ejemplo, así como en C++ se pueden incluir nuevas funciones con ayuda de la instrucción `#include<`*archivo.h*`>`, en LATEX se tienen las instrucciones `\usepackage{`*archivo.sty*`}` y `\documentclass{`*clase.cls*`}` para incluir nuevas instrucciones o modificar el comportamiento de las instrucciones estándar de LATEX. Gracias a estas opciones es posible modificar el formato de un documento fácilmente. Este punto se tratará más a detalle en el siguiente capítulo.

Otra característica importante es la portabilidad de LATEX. Por ejemplo, si se escribe un documento en LATEX en una computadora con un sistema LINUX, es posible abrir tanto el código fuente como el documento final en un sistema WINDOWS sin problema alguno.

Por último, se dice que quien escribe en (LA)TEX debe tener la información organizada para poder escribir. Esto es porque, como se ha mencionado, (LA)TEX es un compilador, y como todo compilador, requiere una mente organizada.

Como es de esperarse, el funcionamiento interno de LATEX también es un poco distinto a los editores de texto; a continuación se explica cómo funciona.

2.2.1. Funcionamiento

A pesar de que LATEX está orientado a crear documentos, no es un editor de textos convencional como lo es MS-WORD, sino que funciona como un compilador. Para comprender dicho funcionamiento considérese el siguiente ejemplo:

> Normalmente, para hacer la publicación de un documento existen tres personas principales: el autor, el diseñador de libros y el cajista. En primer lugar, el autor le entrega a la editorial su escrito. Después, el diseñador de libros investiga cuáles son las intenciones del autor al elaborar el escrito para poder definir la presentación de los títulos, capítulos, citas, ejemplos, fórmulas, etc., además decide el formato del documento (longitud de los renglones, tipo de letra, espaciados, etc.). Por último, el diseñador va con otra persona, el cajista, quien produce este formato para la impresión final del escrito.

De manera análoga, al escribir un documento se tiene en primer lugar al escritor. El diseñador de libros es LATEX, quien da formato al documento, y por último, TEX es el cajista, quien toma el formato y lo aplica al texto para generar el documento final.

De manera más formal, el funcionamiento de LATEX se muestra en la figura 2.1, y se puede dividir en tres etapas:

1. La primera parte requiere de un archivo de texto plano, conocido también como «código fuente», que puede crearse con cualquier editor de textos, guardado con extensión `tex`. En este archivo se escribe tanto el contenido del documento como las instrucciones para crearlo.
2. Para generar el documento se compila el `archivo.tex` con LATEX. Esto genera tres archivos con el mismo nombre que el primero, pero con diferentes extensiones:

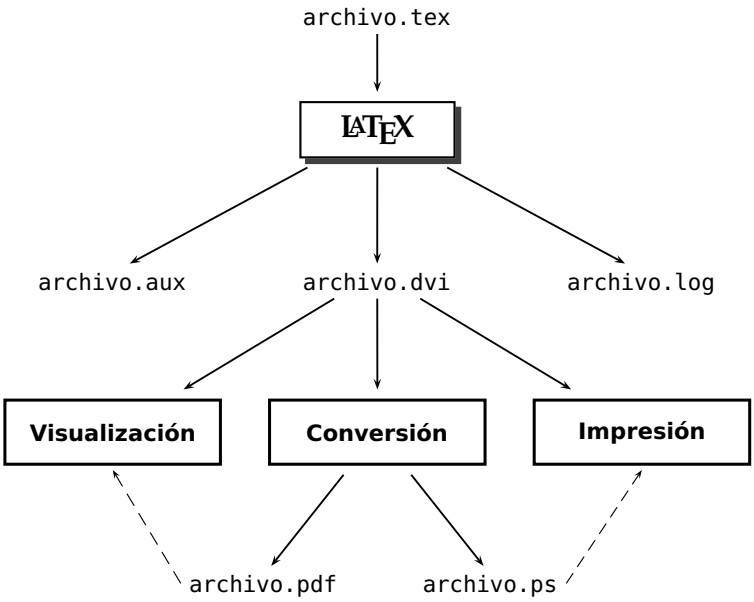


Figura 2.1: Funcionamiento de LaTeX

dvi Este se puede visualizar o imprimir con ayuda de traductores gráficos apropiados.

log Contiene todos los mensajes que salen al momento de la compilación.

aux Contiene información de todas las etiquetas que utiliza LaTeX para producir referencias cruzadas e índices, entre otras cosas.

3. El archivo `dvi` puede utilizarse para lo ya mencionado anteriormente, pero también es posible crear archivos con extensión `ps` o `pdf` para los mismos propósitos. El motivo de esta conversión es que un archivo `ps` es el más adecuado para una impresión profesional, y un archivo `pdf` puede distribuirse fácilmente debido al poco espacio que ocupa. Esta etapa no la realiza LaTeX directamente, sino herramientas adicionales.

Creación de archivos `pdf`

Cuando se compila un documento con LaTeX, simplemente se ejecuta lo siguiente:

```
latex archivo.tex
```

lo que genera el archivo con extensión `dvi`.

Sin embargo, cuando se quiere un archivo `pdf` se recomienda utilizar la instrucción `pdflatex`, la cual evita tener que convertir el archivo `dvi` a `pdf`. Esto es importante, ya que algunos «paquetes» de LaTeX sólo pueden generar archivos `pdf`.

2.3. Creación de documentos

Una vez que se tiene una idea de qué es LaTeX y para qué sirve, es necesario saber cómo elaborar documentos técnicos. Si bien es cierto que trabajar con LaTeX no es tan «cómodo» como trabajar con un procesador del tipo WYSIWYG, la potencia de LaTeX y sus resultados compensan esa incomodidad.

A continuación se da una breve guía de cómo elaborar documentos con LaTeX. No se pretende dar un curso sobre LaTeX, sino explicar las principales instrucciones para crear documentos técnicos. Para mayor detalle en cuanto al funcionamiento de LaTeX se pueden consultar Lamport (1994), Goossens, Mittelbach y Samarin (1994) y Cascales Salinas et al. (2003).

Dentro de la guía, algunas explicaciones vienen acompañadas de un ejemplo, en la parte izquierda se muestra cómo se escribió en el código fuente y la parte derecha se muestra el resultado de la compilación.

2.3.1. Sintaxis básica

Signos de espacio

Los caracteres invisibles, como el espacio en blanco, el tabulador y el final de línea, son tratados por LaTeX como un espacio. Varios espacios en blanco seguidos son tratados como un solo espacio, de igual forma, varias líneas en blanco son tratadas como una sola línea en blanco.

Caracteres reservados

Hay 10 caracteres que para LaTeX son reservados, es decir, no pueden escribirse como cualquier otro, ya que tienen un significado especial, estos se muestran en el cuadro 2.1.

Carácter	Uso en LaTeX	Cómo se escribe
\	Símbolos especiales e instrucciones	`\textbackslash`
{	Comienzo de un grupo	`\{`
}	Final de un grupo	`\}`
%	Comentarios	`\%`
&	Separación entre columnas dentro de una tabla	`\&`
~	Evita separar dos palabras	`\~{}`
$	Inicio o fin del «modo matemático»	`\$`
^	Exponentes	`\^{}`
_	Subíndices	`\_{}`
#	Parámetros para instrucciones	`\#`

Cuadro 2.1: Caracteres reservados dentro de LaTeX

Instrucciones

Las instrucciones de LaTeX se componen de una de las siguientes formas:

1. Comienzan con una barra invertida seguidas de un nombre compuesto por letras y acaban con uno o más espacios en blanco, un carácter especial o un número.

2. Con una barra invertida y un carácter especial.

Después de algunas instrucciones se ignoran los espacios en blanco. Para introducir un espacio se debe poner `{}` o `\` y un espacio.

```
He leído que Knuth distingue a la gente que
trabaja con \TeX\ en \TeX{}nicos y
\TeX  pertos.
```

He leído que Knuth distingue a la gente que trabaja con TEX en TEXnicos y TEXpertos.

Algunas instrucciones necesitan un parámetro que va entre llaves (`{}`), y otras pueden llevar parámetros opcionales que se ponen entre corchetes (`[]`). Por ejemplo:

```
\begin{itemize}
\item \textit{Texto en cursiva}
\item[*] Texto normal
\end{itemize}
```

- *Texto en cursiva*

* Texto normal

También existen unas instrucciones especiales, las cuales se conocen como declaraciones. La estructura de una declaración es la siguiente:

`{\declaración` *texto* `}`

esta declaración afectará al *texto* que se encuentre entre llaves. Si no se ponen las llaves, la declaración afectará al texto que se encuentre después de ésta.

Comentarios

El carácter `%` sirve para hacer anotaciones o comentarios. Todo lo escrito después de este carácter hasta el final de la línea es ignorado al momento de compilar.

Además, este mismo carácter sirve para dividir líneas demasiado largas.

```
Este es un %comentario
ejemplo.\\ Ahora hay otro ejem% se corta
% y se hace otro comentario
plo.
```

Este es un ejemplo.
Ahora hay otro ejemplo.

Unidades de medida

Algunas instrucciones de LaTeX requieren que se especifique alguna unidad de medida, por ejemplo, para indicar el ancho de una columna. Las unidades que LaTeX dispone son las mostradas en el cuadro 2.2.

Unidad	Significado	Muestra
`mm`	milímetro	⌊⌋
`cm`	centímetro	⌊_____⌋
`in`	pulgada	⌊_____⌋
`pt`	punto	‖
`em`	cuadratín	⌊_⌋
`ex`	alto de una x	⌊⌋

Cuadro 2.2: Unidades de medida disponibles

2.3.2. Estructura del código fuente

1. Cuando se compila un archivo en LaTeX se espera una determinada estructura. La primera instrucción indica la clase de documento que se está creando, ésta es:

   `\documentclass`

2. Después se pueden incluir «paquetes». Estos paquetes afectan la composición del documento o agregan nuevas instrucciones. Para incluirlos se utiliza la instrucción:

   `\usepackage`

3. La instrucción `\begin{document}` indica que se inicia el cuerpo del documento. Para finalizar el documento se debe poner la instrucción `\end{document}`; todo lo escrito después de esta instrucción será ignorado por LaTeX.

A la parte comprendida entre `\documentclass` y `\begin{document}` se le llama «preámbulo».

Clases de documentos

Existen varias clases que LaTeX utiliza para dar la estructura del documento. Las clases estándar se muestran en el cuadro 2.3 y pueden agregarse algunas opciones extra que se muestran en el cuadro 2.4. La sintaxis para especificar la clase y sus opciones es la siguiente:

`\documentclass[`*opciones*`]{`*clase*`}`

Clase	Ejemplos de uso
article	Artículos de revistas, trabajos de seminarios, informes pequeños y otros
report	Informes mayores como proyectos de fin de carrera, tesis doctorales, guiones, etc.
book	Libros
letter	Cartas
slide	Transparencias

Cuadro 2.3: Clases estándar para crear documentos

Las opciones predeterminadas para las clases estándar son *letterpage*, *10pt*, *oneside*, *onecolumn* y *final*. La clase report incluye *openany* y la clase book incluye *openright*.

Paquetes

Un documento que necesite incluir gráficos, texto en color o símbolos matemáticos especiales, necesita de instrucciones especiales, las cuales se encuentran en «paquetes». Para incluir estos paquetes se utiliza la instrucción `\usepackage` de la siguiente manera:

`\usepackage[`*opciones*`]{`*paquete*`}`

Opción	Efecto
10pt, *11pt* ó *12pt*	Fija el tamaño de la letra del texto normal.
**paper*	Define el tamaño del papel, puede ser: *letter*, *legal*, *a4*, *a5*, *b5* o *executive*. Por ejemplo, *a4paper*.
titlepage, *notitlepage*	Indica si se debe comenzar una página nueva tras el título o no.
onecolumn, *twocolumn*	Se dispone el documento en una o dos columnas, respectivamente.
oneside, *twoside*	Especifica si se debe generar el documento a una o a dos caras.
final, *draft*	Especifica si se quiere obtener o no una indicación impresa de algunos de los problemas durante la compilación del documento.
openright, *openany*	Hace que los capítulos comiencen o bien, sólo en páginas a la derecha o en la próxima disponible.

Cuadro 2.4: Opciones de clases de documentos

Cada *paquete* tiene sus *opciones* específicas. Las distintas distribuciones de LaTeX incluyen muchos de ellos y otros más pueden obtenerse de internet.

Todos los paquetes deben incluirse en el preámbulo, si se incluyen después de `\begin{document}` causará errores al momento de compilar.

Proyectos grandes

Cuando se crea un documento extenso, lo más recomendable es dividir el trabajo en distintos archivos, por ejemplo, crear un archivo por capítulo. En LaTeX es posible hacerlo con la instrucción

`\include{`*archivo*`}`

Esta instrucción se escribe en donde se desee incluir el texto del *archivo.tex*. El archivo que se incluya no deberá contener preámbulo ni tipo de documento, ni indicar el inicio ni fin de documento, de otra forma no será incluido. La clase de documento definida en el archivo principal del proyecto y el preámbulo afectará a todos los archivos incluidos.

También existe la instrucción

`\includeonly{`*lista de archivos*`}`

donde *lista de archivos* contiene los nombres de los archivos que se desean procesar y tienen que separarse por comas.

La ventaja de esta última instrucción es que sólo se compilarán los archivos de la lista, sin afectar la estructura general del documento, como son los números de página y de capítulo.

Otra instrucción que funciona de manera similar a `\include` es

`\input{`*archivo*`}`

La diferencia es que el *archivo* incluido con `\input` no es controlado con `\includeonly`.

Portada

El título, el autor del documento y la fecha se especifican con las instrucciones:

`\title{`*título del trabajo*`}`
`\author{`*nombre del autor*`}`
`\date{`*fecha*`}`

al inicio del documento. Y con la instrucción

```
\maketitle
```

se creará una portada para el documento. Esta debe llamarse después de las tres anteriores.

La instrucción `\date` es opcional, si no se incluye, LaTeX pondrá la fecha del día en que se procese el archivo. Para no incluir la fecha se tendrá que escribir `\date{}`.

Si son varios los autores del documento, dentro de `\author` se separan con la instrucción `\and`.

2.3.3. Capítulos y apartados

Al escribir un documento se debe tomar en cuenta su estructura para facilitar la lectura y comprensión. Esto se hace por medio de encabezados que definen capítulos, secciones y subsecciones. Para indicarle a LaTeX esta estructura se utilizan las siguientes instrucciones, mostradas de mayor a menor jerarquía:

`\part{`*Título*`}`
`\section{`*Título*`}`
`\subsection{`*Título*`}`
`\subsubsection{`*Título*`}`
`\paragraph{`*Título*`}`
`\subparagraph{`*Título*`}`

Son válidas para las clases `article`, `report` y `book`. Además, se tiene la instrucción

`\chapter{`*Título*`}`

exclusiva para `report` y `book`. La instrucción `\chapter` tiene una jerarquía mayor que `\section` y menor que `\part`.

Con estas instrucciones, LaTeX compondrá automáticamente el tamaño de letra y el espaciado antes y después del título, además de que serán enumerados los cuatro encabezados de mayor jerarquía.

Cuando se crea un apéndice, se tiene la instrucción:

```
\appendix
```

la cual cambia la numeración de las instrucciones `\chapter` o `\section` (según la clase de documento) que se encuentren después de ésta, de número arábigo a letra.

Para incluir la tabla de contenido sólo es necesario agregar la instrucción:

```
\tableofcontents
```

en donde se desee incluir. Para obtener una tabla correcta es necesario compilar el documento tres veces.

Si no se desea enumerar algún título, se tiene que añadir un asterisco en la instrucción antes de las llaves, por ejemplo, `\section*{...}`; sin embargo, el título no aparecerá en la tabla de contenido.

Estilo de las páginas

Cada página del documento tendrá el mismo formato (excepto el inicio de un capítulo y las generadas con la instrucción `\part`), este formato consta básicamente de tres partes:

1. La «cabecera», que es la parte superior de la página en donde se incluye comúnmente el texto del encabezado de mayor jerarquía (capítulo o sección). En ocasiones también se incluye el número de página.
2. El «pie» de la página, que se encuentra en la parte inferior de ésta (no debe confundirse con el espacio para las notas a pie de página); en el pie se incluye en ocasiones el número de página, sino se colocó en la cabecera.
3. El «cuerpo» de la página, que contiene el texto principal del documento.

Al formato de la información de la cabecera y del pie se le conoce como «estilo de página» y puede variar según las guías de estilo de la institución.

En LaTeX existen tres estilos de página básicos, estos son los siguientes:

plain	Imprime los números de página en el centro del pie. Es la opción predeterminada.
headings	En la cabecera se imprime el capítulo y el número de página, y el pie queda vacío.
empty	Tanto la cabecera como el pie quedan vacíos.

Para utilizar alguno de ellos se utiliza la instrucción

`\pagestyle{`*estilo*`}`

la cual define el estilo de página para todo el documento. El estilo *headings* no modifica, sin embargo, el formato para la página donde inicia el capítulo, la cual será del estilo *plain*.

Además de esta instrucción, es posible modificar el formato de una sola página, para ello se tiene la instrucción

`\thispagestyle{`*estilo*`}`

Paginación

Como se mencionó en el primer capítulo, un documento técnico se divide en tres grandes partes: preámbulo, contenido y referencias. La numeración de las páginas en el preámbulo es con números romanos, mientras que en el contenido y referencias se utilizan números arábigos.

Dentro de LaTeX, si se utiliza la clase `book`, se podrá modificar la numeración como se mencionó en el párrafo anterior. Las instrucciones:

`\frontmatter`

```
\mainmatter
\backmatter
```

ayudan a marcar el inicio del preámbulo, del contenido y de las referencias, respectivamente. Estas instrucciones realizan los cambios mencionados en la sección 1.1.2; dichos cambios son los siguientes:

- Con `\frontmatter`, las páginas llevan numeración romana, iniciando en I. Además, la instrucción `\chapter` no muestra la palabra «Capítulo» y los capítulos no se enumeran; sin embargo, las demás instrucciones de encabezados sí aparecen numeradas, lo cual puede mostrar resultados inesperados, por lo que se recomienda utilizar las versiones con asterisco.
- Con `\mainmatter`, la numeración de las páginas vuelve a comenzar, además de utilizar números arábigos. Los encabezados y su respectiva numeración aparecen como se mencionó al principio de esta sección.
- Con `\backmatter`, la numeración de las páginas no se modifica, únicamente se realizan los cambios en los encabezados del igual manera que con la instrucción `\frontmatter`. Sin embargo, colocar los apéndices después de esta instrucción no es lo mejor, ya que no aparecerá ni la palabra «Apéndice» ni la letra correspondiente de éste, por lo que se recomienda colocar `\backmatter` después de los apéndices (a pesar de que pertenecen a la parte de referencias).

2.3.4. Párrafos

LaTeX compone los párrafos de manera automática, tiene predefinido el ancho del párrafo y justifica el texto a ambos márgenes. Además tiene un sistema de espaciado muy complejo, inserta los saltos de línea y los espacios entre las palabras buscando optimizar el contenido de cada párrafo. De igual manera, si es necesario, dividirá alguna palabra que no encaje bien en cada renglón.

Para iniciar un nuevo párrafo es necesario dejar, mínimo, una línea en blanco. Si no se deja esa línea y sólo se baja a la siguiente, LaTeX añadirá un espacio en blanco. Asimismo, se tiene una instrucción para comenzar un nuevo párrafo:

```
\par
```

que debe escribirse cuando se termina éste. Las dos formas son correctas y sólo es necesaria una de ellas.

División silábica

Cuando sea preciso dividir una palabra, LaTeX automáticamente lo hará, pero si no existe esa palabra en el diccionario de LaTeX o se tienen palabras en otros idiomas, se necesitará indicar explícitamente en dónde se podrá hacer la división, esto se puede hacer con la instrucción

```
\-
```

Esta instrucción se utiliza dentro de la palabra e indica que allí se puede dividir la palabra. Por ejemplo:

```
Me parece que esto ha sido: su\-per\-ca\-li\-
fra\-gi\-lis\-ti\-co\-ex\-pia\-li\-do\-so.
```

Me parece que esto ha sido: supercali fragilisticoexpialidoso.

Caracteres especiales

Comillas Para indicar la apertura de comillas se utiliza el símbolo ` o `` (acento grave) dependiendo si son comillas simples o dobles, y para cerrar se utiliza ' o '' (apóstrofo), respectivamente.

```
``Por favor, pulse la tecla `x'.''
```
"Por favor, pulse la tecla `x'."

Guiones LaTeX reconoce cuatro tipos de guiones: el guión, guión largo, la raya y el signo menos. Para obtenerlos se escribe de uno a tres guiones consecutivos, y el cuarto es el signo menos, utilizado en «modo matemático».

```
- -- --- $-$
```
- – — −

Facilidades para otros idiomas

Con el paquete babel se definen tanto el silabeo de las palabras como algunos caracteres especiales del idioma especificado. Este paquete reconoce varios idiomas, entre ellos el español, que se especifica con la opción *spanish*.

Además, con el paquete babel se redefinen los títulos que producen algunas instrucciones de LaTeX que normalmente son en inglés. Por ejemplo, la instrucción `\tableofcontents` mostrará el contenido del documento, pero el título dependerá del idioma: «Table of contents» si es inglés, «Índice general» para español, etc. Lo mismo ocurre para los capítulos, la bibliografía, los apéndices, las figuras y los cuadros.

También existe el paquete inputenc que permite especificar los juegos de asignaciones de caracteres en el documento. Esto es útil, ya que simplifica la escritura del documento. Para el idioma español se introduce la opción *latin1* para escribir los caracteres acentuados tal cual se hace en cualquier otro procesador de textos. De no incluir este paquete, la forma de acentuar, por ejemplo una a, sería `\'a` y con el paquete se escribe á.

2.3.5. Tipografía

Estilo

A lo largo del documento es necesario resaltar alguna parte del texto, por ejemplo, poner un texto en **negrita**, *cursiva*, o VERSALITA. Esto es posible hacerlo en LaTeX con las siguientes instrucciones:

Instrucción	Declaración	Muestra
`\textup`	`\upshape`	Redonda
`\textit`	`\itshape`	*Cursiva*
`\textsl`	`\slshape`	*Inclinada*
`\textsc`	`\scshape`	VERSALITA
`\textmd`	`\mdseries`	Normal
`\textbf`	`\bfseries`	**Negrita**
`\textrm`	`\rmfamily`	Romana
`\textsf`	`\sffamily`	Lineal
`\texttt`	`\ttfamily`	`Monoespaciada`

Pueden hacerse combinaciones entre algunos de estos estilos, por ejemplo, *un texto cursivo y* ***negrito a la vez***. Cabe mencionar que las palabras entre las instrucciones `\texttt` y `\ttfamily` no se dividen, a menos que se especifique con `\-`.

Tamaño

Para modificar el tamaño de la tipografía se tienen las siguientes declaraciones:

Declaración	Muestra
`\tiny`	diminuta
`\scriptsize`	muy pequeña
`\footnotesize`	bastante pequeña
`\small`	pequeña
`\normalsize`	normal
`\large`	grande
`\Large`	mayor
`\LARGE`	muy grande
`\huge`	enorme
`\Huge`	gigante

Otras fuentes de caracteres

También es posible indicarle a LaTeX que utilice otra fuente. La opción predeterminada es la fuente *Computer Roman*, pero existen otras más.

Para cambiar de fuente se tiene que incluir alguno de los siguientes paquetes:

Paquete	Descripción
predeterminado	Utiliza la fuente Computer Roman
`mathpazo`	Utiliza la fuente Palatino
`mathptmx`	Utiliza la fuente Times
`bookman`	Utiliza la fuente Bookman
`newcent`	Utiliza la fuente NewCentury
`utopia`	Utiliza la fuente Utopia
`charter`	Utiliza la fuente Charter
`chancery`	*Utiliza la fuente ZapfChancery*
`helvet`	Utiliza la fuente Helvetica
`avant`	Utiliza la fuente AvantGarde
`courier`	`Utiliza la fuente Courier`

El segundo bloque utiliza fuentes semejantes para el modo matemático. El tercero afecta a la tipografía normal (roman). Utilizando alguna del cuarto bloque se ve afectada la tipografía lineal (sans serif). Y la fuente Courier afecta al texto mecanografiado.

2.3.6. Color

LaTeX dispone del paquete `color` para cambiar el color al texto. Se tienen las instrucciones

`\colorbox{`*color*`}{`*texto*`}`
`\textcolor{`*color*`}{`*texto*`}`

y la declaración

`\color{`*color*`}`

Antes de utilizar un color es necesario definirlo. Dentro del paquete `color` se definen los colores `black`, `white`, `red`, `green`, `blue`, `cyan`, `magenta` y `yellow`.

Para definir un *color*, tiene que escribirse lo siguiente:

`\definecolor{`*nombre*`}{`*modelo*`}{`*parámetros*`}`

Se define el color *nombre* con el *modelo* seleccionado (`gray`, `rgb` o `cmyk`). Los *parámetros* son números entre 0 y 1, y varían según el modelo.

Modelo	Parámetros
`gray`	`{X}`
`rgb`	`{R,G,B}`
`cmyk`	`{C,M,Y,K}`

Por ejemplo, para definir el color rosa se escribe:

```
\definecolor{rosa}{rgb}{1,0.5,0.5}, ó
\definecolor{rosa}{cmyk}{0,0.5,0.5,0}
```

Se recomienda utilizar estas opciones cuando se desea utilizar el documento en la computadora, por ejemplo en una presentación, ya que en la impresión se puede perder la información.

2.3.7. Referencias cruzadas

En los documentos técnicos es común hacer referencia a otra parte del mismo documento; esto es a lo que se llama una referencia cruzada, y puede ser a un capítulo, apartado, figura, tabla, ecuación o a una página. LaTeX proporciona tres instrucciones para las referencias cruzadas:

`\label{`*marcador*`}`
`\ref{`*marcador*`}`
`\pageref{`*marcador*`}`

Para indicar que se puede hacer referencia a cierta parte del documento se introduce la instrucción `\label`. Para hacer la referencia se puede utilizar `\ref` o `\pageref` para hacer referencia a la página. Por ejemplo, esta sección está definida como sigue:

```
\subsection{Referencias cruzadas}\label{sec:Referencias}
```

y para hacer una referencia se escribe:

```
<<Vea la sección~\ref{sec:Referencias}
en la página~\pageref{sec:Referencias}.>>
```

«Vea la sección 2.3.7 en la página 46.»

2.3.8. Notas a pie de página

Para crear una nota a pie de página se utiliza la instrucción:

`\footnote{`*texto de la nota*`}`

Cada nota será numerada automáticamente.

Esta instrucción debe escribirse al término de la palabra donde se desea aparezca la referencia, sin dejar espacios. Por ejemplo:

```
Las notas a pie de página\footnote{Ésta es
una nota a pie de página.} son utilizadas con
frecuencia en los libros.
```

Las notas a pie de página[2] son utilizadas con frecuencia en los libros.

2.3.9. Elementos especiales

Dentro de un documento técnico existen algunas partes donde se requiere de un formato especial para presentar información, por ejemplo: las instrucciones de instalación de algún programa deben ir numeradas, se requiere poner alguna parte de código en algún trabajo de investigación, o se tienen datos de alguna encuesta y se deben mostrar en una tabla. Para crear este tipo de elementos, LaTeX provee de instrucciones especiales, conocidas como «entornos».

De manera general un entorno se define de la siguiente manera:

`\begin{`*entorno*`}`
texto afectado por el entorno
`\end{`*entorno*`}`

Es posible tener un entorno dentro de otro entorno, pero se deben cerrar primero los entornos más internos.

Se presentan a continuación algunos entornos útiles dentro de la documentación técnica.

Listas y descripciones

Existen dos entornos que sirven para crear listas, `itemize` para listas sencillas, `enumerate` para listas enumeradas y el entorno `description` para descripciones. Cada elemento dentro de estos entornos debe ir precedido de la instrucción `\item`. Por ejemplo:

```
\begin{enumerate}
\item Puede mezclar los entornos de listas
a su gusto:
\begin{itemize}
\item Pero podría comenzar a parecer incómodo.
\item Si abusa de ellas.
\end{itemize}
\item Por lo tanto, recuerde:
\begin{description}
\item[Lo innecesario] no va a resultar adecuado
porque lo coloque en una lista.
\item[Lo adecuado,] sí se puede presentar en
una lista.
\end{description}
\end{enumerate}
```

1. Puede mezclar los entornos de listas a su gusto:
   - Pero podría comenzar a parecer incómodo.
   - Si abusa de ellas.
2. Por lo tanto, recuerde:

   Lo innecesario no va a resultar adecuado porque lo coloque en una lista.

   Lo adecuado, sí se puede presentar en una lista.

[2] Ésta es una nota a pie de página.

Pueden anidarse hasta cuatro entornos iguales o seis distintos.

Cuando se utiliza el entorno `description`, la instrucción `\item` requiere de un parámetro opcional el cual se compondrá en negritas, como se observa en el ejemplo anterior.

Alineación horizontal

Se puede justificar un párrafo a la izquierda, a la derecha o al centro. Para dividir los renglones se debe introducir `\\`. La justificación puede hacerse mediante entornos o declaraciones con las siguientes instrucciones:

Entorno	Declaración	Efecto
`flushleft`	`\raggedright`	Texto alineado al margen izquierdo
`flushright`	`\raggedleft`	Texto alineado al margen derecho
`center`	`\centering`	Texto centrado

Por ejemplo:

```
\begin{flushleft}
Este texto está\\ justificado a la izquierda.
\LaTeX{} no intenta forzar que todas las líneas
tengan igual longitud.
\end{flushleft}
```

Este texto está
justificado a la izquierda. LaTeX no intenta forzar que todas las líneas tengan igual longitud.

Citas

El entorno `quote` sirve para hacer citas de otros textos, para dar ejemplos y para resaltar oraciones.

```
Una regla de oro en tipografía para el largo de
los renglones dice:
\begin{quote}
Ningún renglón debe contener más de 66~letras.
\end{quote}
Por esto se suelen utilizar varias columnas
en los periódicos.
```

Una regla de oro en tipografía para el largo de los renglones dice:

> Ningún renglón debe contener más de 66 letras.

Por esto se suelen utilizar varias columnas en los periódicos.

Existe también el entorno `quotation`, que funciona igual que `quote`, pero sangra cada párrafo.

Con las instrucciones `\\` o `\par` se indica el final de una línea.

Código

En algunos documentos es necesario incluir parte del código de algún programa o algoritmo. La tipografía normal en este caso no es la mejor, sino que se recomienda una tipografía «monoespaciada», es decir, todos los caracteres tienen el mismo ancho, semejante a la tipografía de una máquina de escribir.

El entorno `verbatim` y la instrucción

`\verb?`*texto*`?`

sirven para este propósito. El primero compone un bloque de texto de varios renglones tal cual se escribe en el código fuente del documento, mientras que la segunda se utiliza dentro de un párrafo. El caracter `?` en `\verb` puede sustituirse por cualquier otro (excepto el asterisco). Por ejemplo

```
La instrucción \verb|\item |
\begin{verbatim}
10 PRINT "HELLO   WORLD";
20 GOTO  10
\end{verbatim}
```

La instrucción \item

```
10 PRINT "HELLO   WORLD";
20 GOTO  10
```

Existe una variante de estas instrucciones, que muestra los espacios en blanco. Para utilizarla se incluye un asterisco, por ejemplo:

```
La instrucción \verb*|\item |
\begin{verbatim*}
10 PRINT "HELLO   WORLD";
20 GOTO  10
\end{verbatim*}
```

La instrucción \item␣

```
10␣PRINT␣"HELLO␣␣␣WORLD";
20␣GOTO␣␣10
```

Todo el texto dentro de estas instrucciones pierde significado alguno para LaTeX, así que se pueden incluir caracteres reservados o nombres de instrucciones, además de que se incluirán todos los espacios en blanco dejados.

Resumen

Para incluir el resumen o «abstract» de un documento técnico, LaTeX proporciona el entorno `abstract`. Su estructura es:

```
\begin{abstract}
    texto del resumen
\end{abstract}
```

donde *texto del resumen* se reemplaza por el resumen correspondiente. Además, este entorno coloca un título con la palabra «Abstract», «Resumen», etc. según el idioma que se haya especificado.

Tablas

El entorno `tabular` es útil para crear tablas. Este entorno inicia de la siguiente manera:

`\begin{tabular}{`*especificaciones*`}`

donde *especificaciones* define la cantidad de columnas, el ancho de las mismas, si se desea, y con el carácter `|` se indica que va una línea vertical entre dos columnas.

Cada columna tiene su propio justificado: `l` para justificar a la izquierda, `c` para centrar y `r` para justificar a la derecha; dependiendo de la cantidad de `l`, `c` y `r` que se encuentren será la cantidad de columnas de la tabla.

Después de especificar la tabla, se escribe el contenido de la misma. Las columnas se separan con el carácter `&`, y un fin de renglón se indica con `\\`. Si se desea una línea vertical a lo largo de toda la tabla, se debe utilizar la instrucción `\hline` al final del renglón. Por ejemplo:

```
\begin{tabular}{|r|l|} \hline
7C0 & hexadecimal \\ 3700 & octal \\
11111000000 & binario \\\hline\hline
1984 & decimal \\\hline
\end{tabular}
```

7C0	hexadecimal
3700	octal
11111000000	binario
1984	decimal

El ancho de la columna será fijado por la celda más ancha en ésta, sin embargo, es posible especificar el ancho de alguna por medio de la instrucción:

`p{`*tamaño*`}`

en lugar de la letra de justificado.

Si se necesita escribir algún texto que ocupe más de una columna, por ejemplo un encabezado, se tiene la instrucción

`\multicolumn{`*n*`}{`*justificado*`}{`*texto*`}`

donde: *n* es la cantidad de columnas que abarcará el *texto*, y *justificado* puede ser `l`, `r` o `c`. Por ejemplo:

```
\begin{tabular}{p{35mm}r}\hline
\multicolumn{2}{c}{Tercer semestre}\\\hline
\hline Asignatura & Créditos \\\hline
Introducción a la Inteligencia Artificial
& 4.5\\ Ingeniería de Software & 6\\
Sistemas de Transporte de Datos & 4.5 \\
\hline \end{tabular}
```

Tercer semestre	
Asignatura	Créditos
Introducción a la Inteligencia Artificial	4.5
Ingeniería de Software	6
Sistemas de Transporte de Datos	4.5

Gráficos

Cuando se necesite incluir algún archivo gráfico, por ejemplo una fotografía o una ilustración, es necesario cargar el paquete `graphicx`. Este paquete proporciona la instrucción:

`\includegraphics[`*especificaciones*`]{`*archivo*`}`

El *archivo* especifica la ruta del archivo y su extensión; las subcarpetas se dividen con una diagonal (`/`). Las extensiones que LaTeX reconoce son:

- `pdf`, `png`, `jpg`, `jpeg`, `tif` y `tiff` cuando se compila con `pdflatex`
- `ps` y `eps` cuando se compila con `latex`

Dentro de las *especificaciones* se pueden escribir las siguientes opciones (separadas con comas):

`width`	el gráfico se escala al ancho indicado
`height`	el gráfico se escala al alto indicado
`angle`	se rota el gráfico en el sentido de las manecillas del reloj
`scale`	el gráfico es escalado proporcionalmente

Se debe tener cuidado con el orden de las instrucciones, ya que se leerán de izquierda a derecha. Por ejemplo:

```
Se coloca un gráfico:%
\includegraphics[height=9mm]{printer}
Como se observa, aparece sobre el renglón.

Para incluirlo aparte se puede centrar:
\begin{center}
\includegraphics[height=12mm, angle=30]{printer}
\qquad
\includegraphics[angle=30, height=12mm]{printer}
\end{center}
además de que se gira 30~grados.
Es diferente escalar y luego girar,
que girar y luego escalar.
```

Se coloca un gráfico:  Como se observa, aparece sobre el renglón.

Para incluirlo aparte se puede centrar:

además de que se gira 30 grados. Es diferente escalar y luego girar, que girar y luego escalar.

Otra instrucción útil que proporciona el paquete graphicx es:

`\graphicspath{{`*carpeta*$_1$`/},{`*carpeta*$_2$`/},...}`

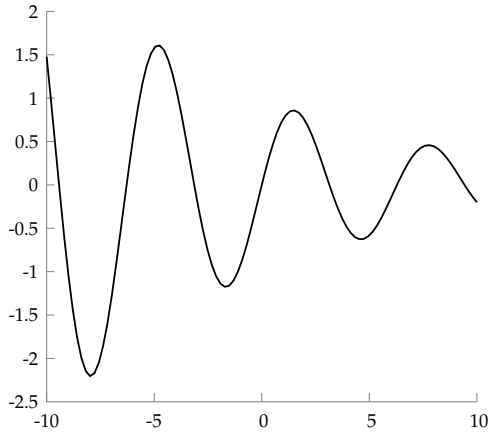
donde carpeta*j* será la ruta en donde se buscarán los archivos incluidos con `\includegraphics`. Todas las carpetas deben terminar con la diagonal, cada una debe estar entre llaves y separadas por comas. Esta instrucción debe colocarse en el preámbulo.

La utilidad de `\graphicspath` es que simplifica el código para incluir un archivo, además de que permite tener todos los archivos más ordenados.

Existen muchos programas de diseño que permiten editar archivos gráficos, entre ellos están: CorelDRAW!, Freehand, Illustrator y Gimp. Sin embargo, existe la opción de crear gráficos directamente con código de LaTeX, utilizando el entorno `picture` o las herramientas PSTricks, una serie de paquetes.

Una de las ventajas de crear un gráfico en LaTeX es que se puede manipular su presentación, por ejemplo, el ancho de línea, la escala y el color; además de que la tipografía será la misma que la del documento. Las herramientas PSTricks son más recomendables que el entorno `picture`, debido a que éste último es limitado en cuanto a las instrucciones que posee. Ambos pueden consultarse en la bibliografía mencionada al principio de este capítulo; aquí sólo se mostrará un ejemplo utilizando las herramientas PSTricks:

```
\begin{pspicture}(0.0,0.0)(1.0,1.0)
% Configuración
\psset{unit=2.7in,xunit=2.7in,yunit=2.3in}
\newpsobject{PSTBorder}{psline}{%
  linewidth=.0015,linestyle=solid,
  linecolor=gray
}
...
% Marco del gráfico
\rput[r](0.1010,0.0840){-2.5}
\PSTBorder(0.1170,0.1822)(0.1320,0.1822)
...
% Gráfica de la función
\PSTSolid(0.1170,0.8656)(0.1170,0.8656)
(0.1254,0.7659)(0.1338,0.6625)(0.1422,0.5596)
(0.1505,0.4615)(0.1589,0.3718)(0.1673,0.2938)
...
\end{pspicture}
```



Elementos flotantes

En ocasiones al momento de compilar el documento, los gráficos y tablas quedan entre dos páginas, lo que ocasiona que el documento contenga mucha información en una de las páginas y poca en la otra, ya que LaTeX no puede dividir estos elementos. Para solucionar este problema, LaTeX cuenta con entornos especiales, llamados «flotantes». Estos entornos pueden moverse del lugar de su definición para que se componga en una sola hoja y presente la información completa.

Los entornos flotantes que LaTeX tiene son: `figure` para componer figuras y `table` para componer cuadros. Dentro de ambos entornos es posible colocar cualquier tipo de información e instrucciones de LaTeX, aunque lo más común es utilizar la instrucción `\includegraphics` para `figure` y el entorno `tabular` para `table`.

Cuando el documento se compondrá a dos columnas, ambos entornos se ajustarán al ancho de la columna, pero si se desea que ocupe el ancho de las dos columnas, tendrá que utilizarse la versión con asterisco de los entornos: `figure*` y `table*`. Las demás opciones de estos entornos funcionan igual para todos.

Sintaxis de un entorno flotante Ambos entornos flotantes tienen un argumento opcional que especifica el lugar donde se desea colocar. Los argumentos permitidos para la colocación están descritos en el cuadro 2.5, pueden escribirse todos a la vez, pero serán leidos de izquierda a derecha.

Argumento	Lugar donde se colocará
`h`	aquí (***h**ere*)
`t`	al inicio de una página (***t**op*)
`b`	al final de una página (***b**ottom*)
`p`	en una página (***p**age*) especial exclusiva para elementos flotantes

Cuadro 2.5: Argumentos de colocación para elementos flotantes

Por ejemplo, el cuadro 2.5 tiene como inicio el siguiente código:

```
\begin{table}[htb]
```

indicando que el cuadro se coloque primeramente en el lugar preciso de su declaración o, en segundo lugar, al inicio de la página o, por último, al final de la misma. Si no se especifican los argumentos de colocación se toman `tbp`.

Otra instrucción útil para estos entornos es la siguiente:

`\caption{`*texto de título*`}`

el cual permite definir un título para el elemento flotante. Este título aparecerá precedido de la palabra «Figura» (para `figure`) o «Cuadro» (para `table`), el número de capítulo y su respectivo número de secuencia.

Con esta información es posible generar un «Índice de figuras» o «Índice de cuadros» por medio de las instrucciones:

```
\listoffigures
\listoftables
```

respectivamente. Funcionan de manera parecida a `\tableofcontents`. Para hacer referencia a esos elementos dentro del documento, se utilizan las instrucciones vistas en la sección 2.3.7.

Tanto la instrucción `\caption` como la instrucción `\label` deben ir dentro del entorno flotante, de preferencia `\label` dentro de `\caption`.

2.3.10. Fórmulas matemáticas

Para escribir fórmulas matemáticas, LaTeX tiene lo que se conoce como «modo matemático» y para acceder a él se tienen instrucciones y entornos especiales. Además se tienen varios paquetes

que complementan la edición de matemáticas, principalmente desarrollados por la AMS, los cuales pueden consultarse en la bibliografía ya citada.

Para escribir alguna fórmula dentro de un párrafo se tienen las instrucciones:

`\(` *fórmula* `\)`,
`$` *fórmula* `$` y
el entorno `math`

En caso que se tengan fórmulas matemáticas o ecuaciones largas, o se desea destacar alguna, es recomendable ponerlas en un párrafo aparte. Para lograr esto, se tiene la instrucción:

`\[` *fórmula* `\]` y
el entorno `displaymath`.

Por ejemplo:

```
Siendo $a$ y \(b\) los catetos y \begin{math}c
\end{math} la hipotenusa de un triángulo
rectángulo, entonces \[c^{2}=a^{2}+b^{2}\]
(Teorema de Pitágoras).
```

Siendo a y b los catetos y c la hipotenusa de un triángulo rectángulo, entonces

$$c^2 = a^2 + b^2$$

(Teorema de Pitágoras).

Si una ecuación se pone dentro de un párrafo tendrá un aspecto distinto al que tiene en un párrafo aparte.

Para enumerar una fórmula se tiene el entorno `equation`. Y para hacer referencia a esta ecuación se utilizan las instrucciones `\label` y `\ref`.

```
\begin{equation} \label{eq:eps}
\epsilon > 0
\end{equation}
De (\ref{eq:eps}) se deduce\...
```

$$\epsilon > 0 \tag{2.1}$$

De (2.1) se deduce...

Las ecuaciones compuestas en un párrafo aparte se centran al ancho del renglón y su número de referencia se coloca del lado derecho. Sin embargo, es posible modificar su presentación con las siguientes opciones de clase:

fleqn	Recorre las ecuaciones a la izquierda.
leqno	Coloca el número de referencia a la izquierda.

El paquete babel con la opción *spanish* proporciona una instrucción útil para el formato del punto decimal. El español que se configura con este paquete utiliza las normas de España, y una de ellas es utilizar una coma para separar los decimales en lugar del punto. La instrucción

```
\decimalpoint
```

cambia la coma decimal por el punto decimal, opción que es utilizada en nuestro país.

Sintaxis en modo matemático

Cuando se edita en modo matemático existen algunas diferencias entre el texto normal y es importante considerarlas. Estas diferencias son:

1. Los espacios en blanco y los cambios de línea no tienen ningún significado.

2. Las letras son consideradas como variables y se escribirán en cursiva.
3. No se pueden incluir caracteres acentuados como en el texto normal, para hacerlo se necesitan instrucciones especiales (véase el cuadro 2.10)
4. Sólo está permitido un párrafo por entorno.

Asimismo, para modificar la tipografía dentro de modo matemático se tienen instrucciones especiales. Estas instrucciones son las siguientes:

Instrucción	Muestra
`\mathnormal{ABCdef123}`	$\mathnormal{ABCdef123}$
`\mathrm{ABCdef123}`	$\mathrm{ABCdef123}$
`\mathbf{ABCdef123}`	$\mathbf{ABCdef123}$
`\mathit{ABCdef123}`	$\mathit{ABCdef123}$
`\mathtt{ABCdef123}`	$\mathtt{ABCdef123}$
`\mathcal{ABC}`	$\mathcal{ABC}$
`\mathbb{ABC}`	$\mathbb{ABC}$
`\mathfrak{ABCabc}`	$\mathfrak{ABCabc}$

Para utilizar las dos últimas instrucciones es necesario incluir el paquete amssymb o amsfonts. Es necesario que todas éstas sean utilizadas en modo matemático.

A continuación se explican otras consideraciones importantes del modo matemático.

Agrupaciones En modo matemático casi todas las instrucciones afectan sólo al carácter que le sigue. Para que la instrucción afecte a más caracteres, estos se deberán encerrar entre llaves ({}). Por ejemplo:

```
\[a^x+y \neq a^{x+y}\]
```

$$a^x + y \neq a^{x+y}$$

Notación matemática

- Los exponentes se indican con el acento circunflejo (^) y los subíndices con el guión bajo (_).
- Existen varios tipos de puntos suspensivos: los alineados a la base del renglón (`\ldots`, $\ldots$), centrados en el renglón (`\cdots`, $\cdots$), verticales (`\vdots`, $\vdots$) y diagonales (`\ddots`, $\ddots$).
- La *n*-ésima raíz se introduce con la instrucción

  `\sqrt[`*n*`]{`*radicando*`}`

  si se omite la *n* se compondrá como raíz cuadrada.
- Una fracción se compone con la instrucción

  `\frac{`*numerador*`}{`*denominador*`}`

- Operadores como el signo de integral (`\int`), la suma (`\sum`) y otros se muestran en el cuadro 2.6. Los límites se indican como exponentes y subíndices.
- Las funciones matemáticas siempre deben ir con tipografía normal, nunca en *cursiva*. Por esta razón se tienen las instrucciones del cuadro 2.7 para escribir los nombres de funciones.

$\bigcap$	\bigcap	$\bigcup$	\bigcup	$\biguplus$	\biguplus	$\bigsqcup$	\bigsqcup
$\bigodot$	\bigodot	$\bigoplus$	\bigoplus	$\bigotimes$	\bigotimes	$\coprod$	\coprod
$\prod$	\prod	$\sum$	\sum	$\int$	\int	$\oint$	\oint
$\bigwedge$	\bigwedge	$\bigvee$	\bigvee				

Cuadro 2.6: Operadores

arccos	\arccos	csc	\csc	ker	\ker	mín	\min
arcsin	\arcsin	deg	\deg	lg	\lg	Pr	\Pr
arctan	\arctan	det	\det	lím	\lim	sec	\sec
arg	\arg	dim	\dim	lím inf	\liminf	sin	\sin
cos	\cos	exp	\exp	lím sup	\limsup	sinh	\sinh
cosh	\cosh	gcd	\gcd	ln	\ln	sup	\sup
cot	\cot	hom	\hom	log	\log	tan	\tan
coth	\coth	ínf	\inf	máx	\max	tanh	\tanh

Cuadro 2.7: Funciones matemáticas

Delimitadores horizontales

Para componer líneas horizontales encima o debajo de una expresión se tienen las instrucciones:

\overline{*fórmula*}
\underline{*fórmula*}

y las instrucciones

\overbrace{*fórmula*}
\underbrace{*fórmula*}

componen llaves horizontales. Por ejemplo:

```
\[\underline{x} \qquad \overline{m+n} \qquad
\underbrace{a+b+\cdots +z}_{26}\]
```

$$\underline{x} \qquad \overline{m+n} \qquad \underbrace{a+b+\cdots+z}_{26}$$

Delimitadores verticales

Para obtener los delimitadores de un tamaño adecuado para la ecuación, es necesario utilizar las instrucciones

\left
\right

seguidos del delimitador. Los distintos delimitadores que tiene LaTeX se muestran en el cuadro 2.8. Por ejemplo:

```
Se tiene la desigualdad:
\[\left\vert\int_0^1f(x)dx\right\vert
\leq\int_0^1\left\vert{}f(x)\right\vert dx\]
Un intervalo semiabierto $\left(\frac{a}{b},
\frac{c}{d}\right]$
```

Se tiene la desigualdad:

$$\left|\int_0^1 f(x)dx\right| \leq \int_0^1 |f(x)|\,dx$$

Un intervalo semiabierto $\left(\frac{a}{b}, \frac{c}{d}\right]$

(	(	)	)	↑	\uparrow	⇑	\Uparrow
[	[	]	]	↓	\downarrow	⇓	\Downarrow
{	\{	}	\}	↕	\updownarrow	⇕	\Updownarrow
⟨	\langle	⟩	\rangle	|	\vert	‖	\Vert
⌊	\lfloor	⌋	\rfloor	⌈	\lceil	⌉	\rceil
/	/	\	\backslash				

Cuadro 2.8: Delimitadores verticales

Se puede sustituir por un punto (`.`) el delimitador que no se desee incluir, ya que LaTeX busca un `\right` por cada `\left` encontrado.

Símbolos

Como es sabido, las fórmulas matemáticas contienen gran variedad de símbolos. Dado que (La)TeX fue creado para esto, se tienen muchísimas instrucciones para generar símbolos. Se presentan a continuación algunos de los más utilizados.

α	\alpha	θ	\theta	o	o	υ	\upsilon
β	\beta	ϑ	\vartheta	π	\pi	ϕ	\phi
γ	\gamma	ι	\iota	ϖ	\varpi	φ	\varphi
δ	\delta	κ	\kappa	ρ	\rho	χ	\chi
ϵ	\epsilon	λ	\lambda	ϱ	\varrho	ψ	\psi
ε	\varepsilon	μ	\mu	σ	\sigma	ω	\omega
ζ	\zeta	ν	\nu	ς	\varsigma		
η	\eta	ξ	\xi	τ	\tau		
Γ	\Gamma[a]	Λ	\Lambda	Σ	\Sigma	Ψ	\Psi
Δ	\Delta	Ξ	\Xi	Υ	\Upsilon	Ω	\Omega
Θ	\Theta	Π	\Pi	Φ	\Phi		

[a] Las demás letras se omitieron debido a que su forma es igual en el alfabeto latino, por ejemplo, la alfa mayúscula es A y puede escribirse como `\mathrm{`A`}`.

Cuadro 2.9: Letras griegas

$\hat{o}$	\hat{o}	$\check{o}$	\check{o}	$\tilde{o}$	\tilde{o}	$\acute{o}$	\acute{o}
$\grave{o}$	\grave{o}	$\dot{o}$	\dot{o}	$\ddot{o}$	\ddot{o}	$\breve{o}$	\breve{o}
$\bar{o}$	\bar{o}	$\vec{o}$	\vec{o}				

Cuadro 2.10: Acentos en modo matemático

$<$	`<`	$>$	`>`	$=$	`=`	$\equiv$	`\equiv`
$\leq$	`\leq`	$\geq$	`\geq`	$\ll$	`\ll`	$\gg$	`\gg`
$\prec$	`\prec`	$\succ$	`\succ`	$\preceq$	`\preceq`	$\succeq$	`\succeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`
$\sim$	`\sim`	$\simeq$	`\simeq`	$\approx$	`\approx`	$\cong$	`\cong`
$\in$	`\in`	$\ni$	`\ni`	$\notin$	`\notin`	$\neq$	`\neq`

Cuadro 2.11: Relaciones binarias

$+$	`+`	$-$	`-`	$\pm$	`\pm`	$\mp$	`\mp`
$\cdot$	`\cdot`	$\div$	`\div`	$\times$	`\times`	$\setminus$	`\setminus`
$\cup$	`\cup`	$\cap$	`\cap`	$\vee$	`\vee`	$\wedge$	`\wedge`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\odot$	`\odot`	$\otimes$	`\otimes`	$\triangleleft$	`\triangleleft`	$\triangleright$	`\triangleright`
$\star$	`\star`	$*$	`\ast`	$\bullet$	`\bullet`	$\diamond$	`\diamond`

Cuadro 2.12: Operadores binarios

$\leftarrow$	`\leftarrow`	$\rightarrow$	`\rightarrow`	$\leftrightarrow$	`\leftrightarrow`
$\longleftarrow$	`\longleftarrow`	$\longrightarrow$	`\longrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`
$\Leftarrow$	`\Leftarrow`	$\Rightarrow$	`\Rightarrow`	$\Leftrightarrow$	`\Leftrightarrow`
$\Longleftarrow$	`\Longleftarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`

Cuadro 2.13: Flechas

Matrices

Para componer matrices se tiene el entorno `array`. Funciona igual que el entorno `tabular`, sólo que `array` debe utilizarse dentro de modo matemático.

Por ejemplo:

```
\[ \left| \begin{array}{*{5}{r}}
1 & 2 & 3 & 4 & 5 \\
6 & 7 & 8 & 9 & 10 \\
11 & 12 & 13 & 14 & 15\\
16 & 17 & 18 & 19 & 20 \\
21 & 22 & 23 & 24 & 25
\end{array}\right|=0\]
```

$$\left| \begin{array}{*{5}{r}} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{array} \right| = 0$$

Ecuaciones largas

Es habitual en matemáticas alinear distintos casos (ecuaciones) posibles agrupándolos, además, con una llave { a la izquierda (por ejemplo, para dar una definición con opciones). Una forma de componer este tipo de estructura es utilizando el entorno `array`.

```
\begin{displaymath}
f\left( x\right)=\left\{%
\begin{array}{cc}
\frac{x^{2}-x-2}{x-2} &
  \textrm{si }x\neq2\\
1 & \textrm{si }x=2
\end{array} \right.
\end{displaymath}
```

$$f\left(x\right)=\left\{\begin{array}{cc}\frac{x^{2}-x-2}{x-2} & \textrm{si }x\neq2\\ 1 & \textrm{si }x=2\end{array}\right.$$

Para ecuaciones que ocupan más de un párrafo o para sistemas de ecuaciones se tiene el entorno `eqnarray`. Funciona como una tabla con tres columnas. Por ejemplo:

```
\begin{eqnarray}
f(x) & = & \cos x \\
f'(x) & = & -\sin x \\
\int_{0}^{x} f(y) \mathrm{d}y & = & \sin x
\end{eqnarray}
```

$$\begin{aligned} f(x) &= \cos x && (2.2)\\ f'(x) &= -\sin x && (2.3)\\ \int_0^x f(y)\mathrm{d}y &= \sin x && (2.4)\end{aligned}$$

Existe el entorno `eqnarray*` que no numera las ecuaciones; dentro de un entorno `eqnarray` para indicar que no se numere la ecuación se escribe la instrucción `\nonumber` antes del fin de renglón (`\\`).

Teoremas y demostraciones

En textos científicos es común incluir enunciados tales como teoremas, corolarios, lemas, definiciones, axiomas, casos, ejemplos, demostraciones, entre otras. Estos enunciados se componen en un párrafo aparte y se enumeran para hacer referencias.

Para componer este tipo de enunciados es preciso declarar los entornos necesarios en el preámbulo. La instrucción

`\newtheorem{`*nombre*`}[`*compartir-contador*`]{`*cabecera*`}[`*ligar-contador*`]`

define un nuevo entorno para componer enunciados. El argumento *cabecera* es el texto que se compone como título del enunciado. El argumento opcional *compartir-contador* sirve para que tanto el entorno *nombre* como el definido como *compartir-contador* compartan la numeración. Y el argumento opcional *ligar-contador* define el contador compuesto por varios niveles, por ejemplo, si se escribe `chapter` la etiqueta mostrará el número del enunciado junto con el número del capítulo. Los enunciados se escriben dentro del entorno *nombre*. Se tienen algunos ejemplos para comprender mejor la explicación:

- En el preámbulo se define lo siguiente:

```
\newtheorem{propC}{Proposición}[chapter]

\newtheorem{propS}{Proposición}[section]

\newtheorem{prop}{Proposición}
```

  Utilizando estos entornos se obtiene lo siguiente:

```
\begin{propC}
El conjunto de los números primos es
infinito.
\end{propC}
\begin{propS}
El conjunto de los números primos es
infinito.
\end{propS}
\begin{prop}
El conjunto de los números primos es
infinito.
\end{prop}
```

Proposición 2.1 *El conjunto de los números primos es infinito.*

Proposición 2.3.1 *El conjunto de los números primos es infinito.*

Proposición 1 *El conjunto de los números primos es infinito.*

- Para ligar los contadores se define como sigue:

```
\newtheorem{definicion}{Definición}
\newtheorem{proposicion}[definicion]{Proposición}
```

Utilizando estos entornos se obtiene lo siguiente:

```
\begin{definicion}
Sea $\{X_t\}_{t\in T}$ un proceso estocástico
con espacio de estados $(E,\mathcal{E})$,\...
\end{definicion}
\begin{proposicion}
Sea $T$ un espacio métrico separable, $E$
espacio métrico y $\{X_t\}_{t\in T}$
un proceso estocástico\...
\end{proposicion}
```

Definición 1 *Sea $\{X_t\}_{t\in T}$ un proceso estocástico con espacio de estados $(E,\mathcal{E})$,...*

Proposición 2 *Sea T un espacio métrico separable, E espacio métrico y $\{X_t\}_{t\in T}$ un proceso estocástico...*

Además, todos los enunciados tienen un argumento opcional *identificación* que permite citar el autor original o el nombre con el cual se conoce al enunciado:

`\begin{`*nombre*`}[`*identificación*`]`

Esta *identificación* se compone entre paréntesis y en negrita, en la cabecera del enunciado. Por ejemplo:

```
\begin{teorema}[Grassman]
Sean $F$ y $G$ dos subespacios vectoriales de
$E$ y supongamos que la dimensión de $E$ es
finita. Entonces $F$, $G$, $F\cap G$ y $F+G$
son todos de dimensiónfinita y $$\dim F+\dim G=
\dim(F+G)+\dim(F\cap G).$$
\end{teorema}
```

Teorema 1 (Grassman) *Sean F y G dos subespacios vectoriales de E y supongamos que la dimensión de E es finita. Entonces F, G, $F\cap G$ y $F+G$ son todos de dimensiónfinita y*

$$\dim F + \dim G = \dim(F+G) + \dim(F\cap G).$$

Para hacer una referencia a los enunciados sólo se tiene que incluir la instrucción `\label` dentro del entorno.

2.3.11. Bibliografía

Las citas bibliográficas son referencias cruzadas hacia una lista de referencias bibliográficas. De igual manera que funcionan las instrucciones `\label` y `\ref` para las referencias cruzadas, se tienen las instrucciones `\bibitem` y `\cite` para las citas bibliográficas.

La instrucción

`\cite[`*nota*`]{`*clave*`}`

introduce una cita bibliográfica hacia la referencia bibliográfica que tiene *clave* por identificador. El parámetro *nota* permite componer una nota como parte de la cita bibliográfica, por ejemplo para dirigir al lector hacia un apartado o página específicos de la obra.

```
Las estructuras de grafo~\cite[pág.~188]{loewe}
han sido interpretadas en~\cite{atupa} como
álgebras parciales unarias.
```

Las estructuras de grafo (Löw93, pág. 188) han sido interpretadas en (BR+96) como álgebras parciales unarias.

Cuando se tienen que hacer dos o más citas bibliográficas consecutivas es preciso incluir todas las citas dentro de una misma instrucción `\cite` separadas por comas.

Como complemento de las citas bibliográficas es necesaria una bibliografía. Existe el entorno `thebibliography` que compone la bibliografía o lista de referencias bibliográficas. Los elementos dentro de este entorno se especifican con la instrucción

`\bibitem[`*etiqueta*`]{`*clave*`}` *texto de la referencia*

donde *etiqueta* es la etiqueta de la cita bibliográfica, creada según algún sistema de citación, *clave* es la clave de para hacer referencias con `\cite` y *texto de la referencia* es el texto de la referencia bibliográfica escrita según algún estilo bibliográfico.

Bases de datos bibliográficos con BibTeX

Muchos de los datos bibliográficos recolectados pueden ser necesarios para distintos trabajos, así que una cuidadosa recolección y un registro correcto de esos datos pueden ser buena inversión.

BibTeX es un sistema de marcado que permite generar bases de datos bibliográficos. Cuenta con instrucciones específicas para definir las referencias bibliográficas y permite definir nuevos estilos bibliográficos. BibTeX es un complemento de LaTeX para generar la bibliografía de algún documento. Y es una mejor opción que el entorno `thebibliography`.

BibTeX maneja, al igual que LaTeX, archivos de texto plano, y la extensión debe ser `bib`.

Sintaxis de BibTeX El proceso para crear la bibliografía con ayuda de BibTeX puede verse en la figura 2.2. Para que la bibliografía se incluya correctamente, primero se compila el documento con LaTeX, después se compila el mismo archivo pero con el programa `bibtex` (generando un archivo con extensión `bbl`) y por último se debe compilar el documento con LaTeX para que se incluya la bibliografía.

Dentro de una base de datos bibliográficos en BibTeX, las referencias tienen la siguiente forma:

```
@categoría { clave,
      entrada = "texto",
      entrada = "texto",
            ⋮
      entrada = "texto" }
```

donde *clave* es el identificador usado para hacer citas bibliográficas mediante la instrucción `\cite`. El *texto* asociado a cada *entrada* tiene que ir delimitado por llaves de apertura y cierre o entre comillas dobles (`"`). Al final de cada *texto* se debe poner una coma, excepto en el último.

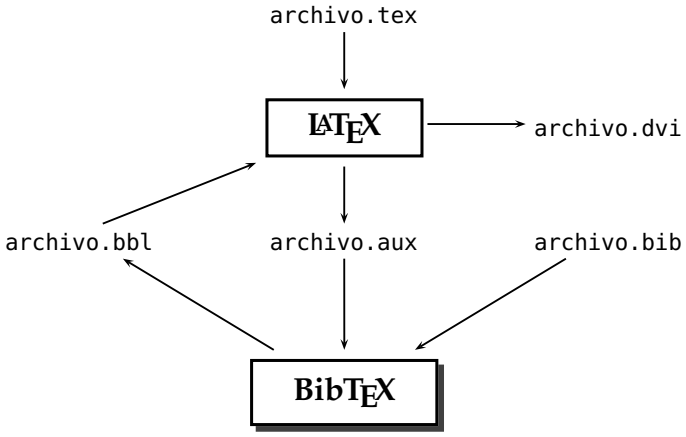


Figura 2.2: Funcionamiento de BibTeX

Las categorías estándares de BibTeX son las siguientes:

article Un artículo publicado en una revista. Entradas obligatorias: *author*, *title*, *journal* y *year*. Entradas optativas: *volume*, *number*, *pages*, *month* y *note*.

book Un libro. Entradas obligatorias: *author* o *editor*, *title*, *publisher* y *year*. Entradas optativas: *volume* o *number*, *series*, *address*, *edition*, *month* y *note*.

booklet Un documento impreso y encuadernado pero que no ha sido publicado. Entrada obligatoria: *title*. Entradas optativas: *author*, *howpublished*, *address*, *month*, *year* y *note*.

inbook Una parte de un libro, como por ejemplo un capítulo o una serie de páginas consecutivas. Entradas obligatorias: *author* o *editor*, *title*, *chapter* y/o *pages*, *publisher* y *year*. Entradas optativas: *volume* o *number*, *series*, *type*, *address*, *edition*, *month* y *note*.

incollection Una parte de un libro con un título propio. Entradas obligatorias: *author*, *title*, *booktitle*, *publisher* y *year*. Entradas optativas: *editor*, *volume* o *number*, *series*, *type*, *chapter*, *pages*, *address*, *edition*, *month* y *note*.

inproceedings Un artículo publicado en la miscelánea de un congreso. Entradas obligatorias: *author*, *title*, *booktitle* y *year*. Entradas optativas: *editor*, *volume* o *number*, *series*, *pages*, *address*, *month*, *organization*, *publisher* y *note*.

manual Un manual de documentación técnica. Entrada obligatoria: *title*. Entradas optativas: *author*, *organization*, *address*, *edition*, *month*, *year* y *note*.

masterthesis Una tesis de licenciatura. Entradas obligatorias: *author*, *title*, *school* y *year*. Entradas optativas: *type*, *address*, *month* y *note*.

misc Un trabajo que no se ajusta a ninguna de las categorías. No tiene entradas obligatorias. Entradas optativas: *author*, *title*, *howpublished*, *month*, *year* y *note*.

phdthesis Una tesis doctoral. Entradas obligatorias: *author*, *title*, *school* y *year*. Entradas optativas: *type*, *address*, *month* y *note*.

proceedings La miscelánea de un congreso. Entradas obligatorias: *title* y *year*. Entradas optativas: *editor*, *volume* o *number*, *series*, *month*, *organization*, *publisher* y *note*.

techreport Un reporte técnico o de investigación. Entradas obligatorias: *author*, *title*, *institution* y *year*. Entradas optativas: *type*, *number*, *address*, *month* y *note*.

unpublished Un trabajo inédito. Entradas obligatorias: *author*, *title* y *note*. Entradas optativas: *month* y *year*.

Si no se incluyen las entradas obligatorias, durante la compilación con BibTeX, se generará un error.

Las entradas estándares de información se muestran en el cuadro 2.14. Para una explicación más detallada de estas entradas puede consultarse Cascales Salinas et al. (2003, pág. 259).

address	*annote*	*author*	*booktitle*
chapter	*crossref*	*edition*	*editor*
howpublished	*institution*	*journal*	*key*
month	*note*	*number*	*organization*
pages	*publisher*	*school*	*series*
title	*type*	*volume*	*year*

Cuadro 2.14: Entradas estándar de información

Dentro de las entradas *author* y *editor* el nombre debe comenzar por los apellidos y luego el nombre o nombres separados mediante una coma. Cuando se tienen varios autores, se separan con la palabra `and`.

Uso de la base de datos bibliográficos en LaTeX

Una vez elaborada la base de datos bibliográficos con BibTeX, se puede ocupar en LaTeX. La instrucción

`\bibliography{`*lista de archivos*`}`

buscará en *lista de archivos* las citas hechas en el documento. La lista debe estar separada con comas y no se debe incluir la extensión.

Durante la composición de la bibliografía, únicamente se incluirán aquellas que hayan sido citadas con la instrucción `\cite`. Si se desea incluir alguna otra referencia no citada en el documento, se tiene la instrucción

`\nocite{`*lista de claves*`}`

la cual incluirá las referencias correspondientes a las claves de la *lista de claves*. Pero si se desean incluir todas las referencias de los archivos `bib`, se sustituye la *lista de claves* por un asterisco ($*$).

Estilo bibliográfico Para seleccionar un estilo bibliográfico se utiliza la instrucción

`\bibliographystyle{`*estilo*`}`

donde *estilo* es uno de los siguientes:

plain	Sistema numérico de citación y estilo bibliográfico numérico. Las referencias bibliográficas son ordenadas alfabéticamente.
unsrt	Sistema numérico de citación y estilo bibliográfico numérico. Las referencias bibliográficas mantienen el orden que tienen dentro de la base de datos.
alpha	Sistema autor-fecha abreviado de citación y estilo bibliográfico alfabético.
abbrv	Sistema autor-fecha abreviado de citación y estilo bibliográfico alfabético abreviado.

Además de estos estilos, las distribuciones de LaTeX proveen de muchos más. Uno de los más utilizados es el estilo «chicago» y para incluirlo se necesita cargar el paquete achicago y escoger el estilo *achicago* para la bibliografía.

Un ejemplo de cómo utilizar BibTeX es el siguiente. Primero, se define la referencia de un libro con el siguiente código:

```
@book{TeXBook,
author={Knuth, Donald Ervin},
title={The TeXbook},
publisher={Addison-Wesley},
year={1986}
}
```

Para hacer una cita, dentro de LaTeX se utiliza la instrucción `\cite`:

```
\...Donald E. Knuth, quien dice
al comienzo de su libro
\textit{The \TeX book}, \cite{TeXBook}:
```

...Donald E. Knuth, quien dice al comienzo de su libro *The TeXbook*, (Knuth 1986):

2.3.12. Índices

Un índice, como se mencionó en el capítulo anterior, permite localizar información específica dentro del documento, y debe dar más información que la tabla de contenido.

Para generar índices se tiene un programa externo a LaTeX: MakeIndex. El funcionamiento de MakeIndex se muestra en la figura 2.3. Como se observa, funciona de manera similar a BibTeX.

Preparación de índices con MakeIndex

Para poder utilizar el programa MakeIndex se debe incluir el paquete makeidx. Una vez incluido, se tiene que incluir la instrucción

```
\makeindex
```

en el preámbulo, e incluir la instrucción

```
\printindex
```

en donde se quiera colocar el índice.

Cada entrada del índice se define con la instrucción

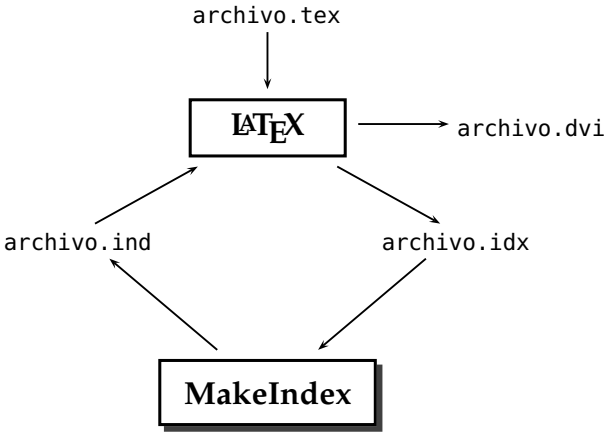


Figura 2.3: Funcionamiento de MakeIndex

`\index{`*texto*`!`*subtexto*`!`*subsubtexto*`}`

Dependiendo del nivel de detalle, esta instrucción puede omitir *subtexto* y *subsubtexto*. Por ejemplo:

Página 63:	`\index{bibliografías!concepto}`	bibliografías
Página 64:	`\index{bibliografías!estilo!descriptivo}`	concepto, 63
Página 64:	`\index{bibliografías!estilo!normalizado}`	estilo
Página 71:	`\index{bibliografías!estilo!descriptivo}`	descriptivo, 64, 71, 89
Página 81:	`\index{bibliografías!ordenación}`	normalizado, 64, 85
Página 85:	`\index{bibliografías!estilo!normalizado}`	ordenación, 81
Página 69:	`\index{bibliografías!estilo!descriptivo}`	

Para especificar el rango de páginas en donde se trata el concepto, la instrucción `\index` se complementa de la siguiente manera:

`\index{`*texto*`|(}`

para indicar que inicia la definición de la entrada *texto*, y con la instrucción

`\index{`*texto*`|)}`

se indica que termina dicha definición. Por ejemplo:

Página 63:	`\index{bibliografías!concepto|(}`	bibliografías
Página 64:	`\index{bibliografías!concepto|)}`	concepto, 63–64
Página 64:	`\index{bibliografías!estilo!descriptivo|(}`	estilo
Página 66:	`\index{bibliografías!estilo!descriptivo|)}`	descriptivo, 64–66, 71
Página 71:	`\index{bibliografías!estilo!descriptivo}`	

Por último, para incluir una referencia cruzada dentro del índice se tiene el complemento

`\index{`*texto*`|see{`*referencia*`}}`

Por ejemplo:

```
Página 121:  \index{Valor presente neto}
Página 122:  \index{\textit{VPN}
                    |see{Valor presente neto}}
```

Valor presente neto, 121
VPN, *véase* Valor presente neto

2.4. Presentaciones

También es posible crear presentaciones con LaTeX, existen varios paquetes que generan presentaciones de alta calidad. Con algunos paquetes se puede obtener una versión impresa de la presentación o electrónica para ser proyectada en pantalla, generalmente en formato `pdf`; ejemplos de estos paquetes son: `pdfslide`, `pdfscreen` y `web`, además de la clase `prosper`.

A continuación se da una breve explicación de cómo crear presentaciones con la clase `prosper`.

2.4.1. Opciones de la clase prosper

La clase `prosper` se utiliza como las clases estándar de LaTeX, pero con diferentes opciones, además de que incluye los paquetes `seminar`, `hyperref` y `graphicx`. Las opciones de esta clase se muestran en el cuadro 2.15; las opciones predeterminadas son: *ps*, *final*, *total*, *slideBW*, *nocolorBG*, *noaccumulate*.

Opción	Efecto
ps, *pdf*	Se genera un archivo PostScript o en formato `pdf`, respectivamente.
final, *draft*	Con *final* se incluye en el pie de la diapositiva la información de `\slideCaption` y con *draft* se incluye el nombre del archivo, el título, el autor y la fecha de compilación.
total, *nototal*	Con *total* aparece en el pie el número de diapositiva junto con el total de diapositivas y con *nototal* sólo se incluye el número de diapositiva correspondiente.
slideBW, *slideColor*	Componen la presentación a blanco y negro o a color, respectivamente.
nocolorBG, *colorBG*	Con *nocolorBG* el fondo de las diapositivas será blanco, mientras que con *colorBG* el fondo dependerá del «estilo» elegido.
accumulate, *noaccumulate*	Estas opciones son útiles para manipular las animaciones, *noaccumulate* muestra las animaciones y *accumulate* muestra toda la información de la diapositiva sin animaciones.

Cuadro 2.15: Opciones de la clase prosper

Otra opción que se debe especificar es el «estilo». El estilo define los colores del texto, la tipografía y el fondo de las diapositivas. Algunos estilos disponibles son los siguientes:

alienglow	autumn	azure	contemporain
darkblue	frames	lignesbleues	nuancegris
troispoints	gyom	pascal	rico

2.4.2. Compilación

Para obtener el archivo final de la presentación utilizando la clase prosper, se tiene un proceso diferente al mostrado al inicio del capítulo. Este proceso se muestra en la figura 2.4.

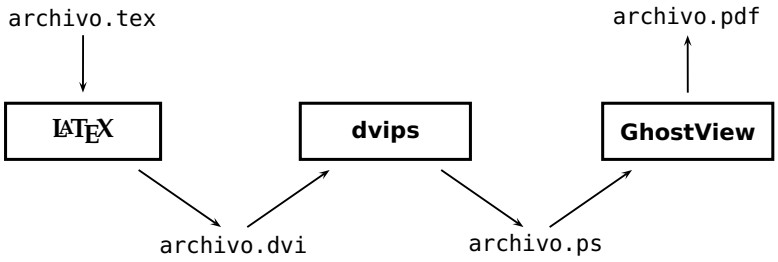


Figura 2.4: Creación de una presentación con prosper

Una vez que se crea el archivo `dvi` debe convertirse a formato `ps`, ya que una compilación directa a `pdf` genera errores.

2.4.3. Estructura del archivo fuente

Cuando se utiliza la clase prosper, la estructura del archivo de la presentación es más simple que la de un documento común. En el preámbulo, además de las instrucciones comunes (tipo de documento, inclusión de paquetes y definición de instrucciones) se pueden incluir las siguientes:

`\title{`*Título de la presentación*`}`
`\author{`*Autores*`}`
`\subtitle{`*Subtítulo*`}`
`\email{`*E-mails*`}`
`\institution{`*Institución*`}`
`\Logo(`x,y`){`*Logotipo*`}`
`\slideCaption{`*Leyenda*`}`
`\displayVersion`
`\DefaultTransition{`*Transición*`}`

Las instrucciones `\title` y `\author` son las únicas necesarias para generar la portada y funcionan igual que en las demás clases. Con `\Logo` se colocará el archivo *Logotipo* con extensión `eps` o `ps` en la posición (x, y) (la coordenada $(0, 0)$ está en la esquina inferior izquierda de la diapositiva); si no se incluye la coordenada, el *Logotipo* se colocará en la posición que le asigne el estilo elegido.

La instrucción `\slideCaption` define la información que se presentará en el pie de la diapositiva, y con la declaración `\displayVersion` se colocará en el pie la información que se muestra al compilar con la opción *draft*.

La instrucción `\DefaultTransition` define la transición predeterminada de todas las diapositivas. *Transición* puede ser alguna de las mencionadas en el cuadro 2.16.

Opción	Efecto
`Split`	Dos líneas limpian la pantalla para mostrar la nueva diapositiva, como unas cortinas.
`Blinds`	Similar a `Split`, pero con varias líneas.
`Box`	Un cuadro crece desde el centro para mostrar la nueva diapositiva.
`Wipe`	Una línea limpia la pantalla para mostrar la nueva diapositiva.
`Dissolve`	La pantalla se disuelve para mostrar la nueva diapositiva.
`Glitter`	Similar a `Dissolve`, pero esta opción inicia en un lado de la pantalla.
`Replace`	La pantalla es remplazada con la nueva diapositiva. Esta es la opción predeterminada.

Cuadro 2.16: Transiciones entre diapositivas

Dentro del entorno `document`, la primera instrucción que debe aparecer es `\maketitle` si se desea crear la portada, y la información de cada diapositiva se define dentro de un entorno `slide` de la siguiente manera:

`\begin{`*slide*`}[`*Transición*`]{`*Título de la diapositiva*`}`
contenido de la diapositiva
`\end{`*slide*`}`

La *Transición* es opcional, de no indicarse se toma la definida en `\DefaultTransition`.

Con la instrucción:

`\part[`*Transición*`]{`*Título de la sección*`}`

se crea una diapositiva en blanco únicamente con el texto *Título de la sección*.

Para elaborar animaciones se tiene la instrucción:

`\overlays{`*n*`}{...}`

donde *n* es la cantidad de etapas que contendrá la diapositiva para presentar toda la información. El segundo argumento de esta instrucción es comúnmente un entorno `slide`.

Además de `\overlays` existe el entorno `itemstep` que ayuda a crear la animación. Este entorno funciona igual que `itemize`, sólo que va mostrando los ítems etapa por etapa.

Por último, se tienen las siguientes instrucciones:

`\fromSlide{`*p*`}{`*Texto*`}`
`\onlySlide{`*p*`}{`*Texto*`}`
`\untilSlide{`*p*`}{`*Texto*`}`

```
\FromSlide{p}
\OnlySlide{p}
\UntilSlide{p}
```

que también ayudan a construir las animaciones. Con `\fromSlide` el *Texto* aparecerá a partir de la etapa *p*, `\onlySlide` sólo muestra el *Texto* en la etapa *p* y `\untilSlide` mostrará el *Texto* hasta la etapa *p*. Las últimas tres declaraciones funcionan de igual manera que su respectiva instrucción.

Cabe mencionar que para obtener las animaciones en formato `pdf`, en `\documentclass` se debe utilizar la opción *pdf*; además para utilizar las últimas instrucciones que ayudan a construir la animación se deben especificar antes las etapas con `\overlays`, de otra forma sólo aparecerá una sola diapositiva.

2.4.4. El paquete hyperref

El paquete hyperref es muy útil para crear enlaces a otras partes del documento, de manera similar a los «links» de una página web, lo cual permite poder construir una barra de navegación dentro de la presentación, además de que es posible incluir elementos multimedia; sin embargo, no se tratará en este trabajo, principalmente porque este paquete funciona correctamente con el programa ACROBAT de ADOBE. Para más información se puede consultar Cascales Salinas et al. (2003) o la documentación del paquete.

2.5. Herramientas complementarias

Como se ha podido observar, LaTeX cuenta con varios paquetes que hacen de LaTeX más que un procesador de textos, además de que existen programas adicionales que hacen tareas en las que LaTeX está más limitado.

No sólo es posible crear con LaTeX documentos que serán impresos, como un manual de usuario o un trabajo de investigación, sino que es posible crear sitios web, carteles y otros documentos técnicos. Además es posible utilizar distintos alfabetos, por ejemplo, se puede escribir en alemán, árabe, griego, hebreo, ruso, entre otros; inclusive se pueden crear partituras con un complemento que se distribuye junto con el compilador.

Para un estudio más detallado de estas extensiones de LaTeX, además de consultar la bibliografía ya citada, en cada distribución de LaTeX se incluye una carpeta (comúnmente llamada `doc`) con la documentación de casi todos los paquetes incluidos, y en algunos casos se incluye también el manual de uso del paquete.

Además, en internet es posible obtener gran cantidad de material sobre (LA)TeX. Existen sitios como TeX User Group (`www.tug.org`), CervanTeX (`www.cervantex.org`) y Comprehensive TeX Archive Network (`www.ctan.org`) los cuales contienen noticias, manuales, paquetes y software para trabajar con (LA)TeX. El sitio CTAN es el más adecuado si se necesita alguna clase o paquete para (LA)TeX; todo el sitio está bien organizado, cualquier archivo y directorio contiene una descripción de su contenido y cuenta con una gran cantidad de «mirrors» para descargar los archivos, entre ellos existe uno en la Facultad de Ciencias de la U.N.A.M. En el apéndice B se agregan las direcciones electrónicas del software incluido en el CD de este trabajo.

Capítulo 3

Programación de plantillas

De manera general un documento se forma de tres elementos:

1. Contenido
2. Estructura
3. Formato

Por «contenido» se entiende al conocimiento que se transmite por medio del documento, la «estructura» indica el orden en que se debe presentar el contenido y el «formato» se ocupa de cómo mostrar la información, ya sea en pantalla o impresa. Si alguno de estos elementos no está presente, el documento no logrará su objetivo.

En el primer capítulo se explicó cómo colocar el contenido del documento de manera apropiada para el lector; también se explicó cuál es la estructura de los trabajos más utilizados en la carrera de Matemáticas Aplicadas y Computación. Lo que corresponde ahora es hablar del formato utilizado para crear documentos técnicos.

Dentro de la documentación técnica se tienen las «plantillas» para darle el formato al documento. Una plantilla especifica el diseño de los distintos elementos del documento, como son: el tamaño de papel, los márgenes de la página, la tipografía para el texto normal y para los encabezados, la información de las cabeceras y los pies y la colocación de las listas, los gráficos, de las tablas y otros elementos de este tipo. Esta información se toma de la guía de estilos definida en la etapa de diseño (sección 1.2.2), durante la elaboración del documento técnico.

Las plantillas son utilizadas dentro de los distintos programas de cómputo para elaborar documentos, por ejemplo, en MS-WORD se tienen «plantillas» y «estilos», y en FRAMEMAKER se tienen las «plantillas de esquema».

Utilizar una plantilla es útil, ya que agiliza la creación del documento, y quienes se dedican a capturarlo se pueden olvidar de cómo colocar la información, simplemente aplican el estilo correspondiente y el programa utilizará el formato especificado en la plantilla. Como se mencionó en el primer capítulo, la información del documento debe presentarse de una manera atractiva para el lector y de forma que pueda localizar fácilmente la información que necesita. Si la plantilla está mal diseñada, el documento perderá efectividad.

Actualmente, en muchos lugares donde se maneja una gran cantidad de documentos, como la IEEE, ISO y varias universidades en el mundo, así como en las editoriales más reconocidas a nivel

mundial, han diseñado sus plantillas para facilitar el trabajo del escritor. Las plantillas están diseñadas para MS-WORD, TEX y LATEX. MS-WORD es el procesador de textos más común, sin embargo, cuando el documento contiene notación matemática, todos recomiendan utilizar TEX o LATEX. La mayoría de estas plantillas se pueden obtener de internet.

Sin embargo, el conocimiento (y, por consecuencia, el uso) de (LA)TEX en México es escaso. En pocas instituciones se utiliza TEX o LATEX para elaborar documentos, y son menos las que cuentan con plantillas propias. Únicamente la Universidad de las Américas Puebla[1] cuenta con una plantilla para elaborar las tesis de los departamentos de Física y Matemática, además la Facultad de Ciencias de la U.N.A.M. tiene una plantilla para elaborar presentaciones[2].

Es precisamente la programación de plantillas el punto principal de este trabajo. Las cinco plantillas que se programarán serán una herramienta útil para todos aquellos que utilizan LATEX, ya que se trata de plantillas para documentos que se utilizan con frecuencia en las carreras de las áreas de Matemáticas e Ingenierías. Esto permitirá que el escritor ocupe su tiempo en lo que realmente es importante, es decir, el contenido del documento, dejando las cuestiones de diseño a LATEX y a la plantilla correspondiente.

En las secciones siguientes se da una explicación de cómo se puede modificar el formato que LATEX utiliza para componer documentos y se explica también cómo crear plantillas para este lenguaje. Se termina este capítulo con el análisis de las cinco plantillas que se programarán.

3.1. Plantillas en LATEX

En el segundo capítulo se mencionó que LATEX es útil para elaborar documentos técnicos, y de igual manera que los demás programas de este tipo, es posible crear plantillas para los documentos de LATEX.

LATEX utiliza dos tipos de archivos con los que se puede crear una plantilla: las clases (archivos con extensión `cls`) y los paquetes (con extensión `sty`). Para crear algún archivo de este tipo se utilizan instrucciones de LATEX y TEX especiales, y se dice que se «programa la plantilla». El término de programación se utiliza, ya que como se mencionó en el capítulo anterior, (LA)TEX es un lenguaje de programación y funciona de manera similar que HTML, es decir, es un lenguaje de marcado.

A continuación se explican más a detalle las instrucciones para programar dentro de LATEX.

3.1.1. Programación en LATEX

Como se ha visto en el capítulo anterior, existen varias maneras de dar formato a un documento en LATEX, por ejemplo, instrucciones que modificarán todo lo que se encuentre después de ellas (como `\itshape`, que colocará el texto en cursiva), instrucciones que modificarán sólo una parte (como `\textbf{`*Texto*`}` que cambiará el *Texto* en negritas), otras más que realizan una tarea específica (como `\maketitle` que compone la portada), así como entornos que modifican una cantidad mayor de objetos (como `center`). Pero es posible también definir nuevas instrucciones y entornos; esto se explica a continuación.

[1] Se puede consultar la página `http://www.udlap.mx/~ma108907/latex` para mayor información.

[2] En la página `http://www.fciencias.unam.mx:8085/servlets/ciencias/area/31/section/80` se puede descargar dicha plantilla; existen otros proyectos, pero no todos están disponibles.

Nuevas instrucciones

Para definir nuevas instrucciones o modificar el funcionamiento de alguna existente, LaTeX tiene las siguientes instrucciones:

`\newcommand*{\`*Nombre*`}[`*NúmArg*`]{`*Definición*`}`
`\renewcommand*{\`*Nombre*`}[`*NúmArg*`]{`*Definición*`}`
`\providecommand*{\`*Nombre*`}[`*NúmArg*`]{`*Definición*`}`

Con `\newcommand*` se define una nueva instrucción llamada *Nombre*, que hará lo que *Definición* indique. Por ejemplo, para dar un formato especial al nombre de los programas de cómputo se realiza lo siguiente:

```
\newcommand*{\software}[1]{\textit{#1}}

Existen varios programas que sirven para
elaborar documentación técnica, entre ellos
se tiene: \software{FrameMaker},
\software{Ventura Publisher},
\software{FreeHand}, \software{MS-Word},\...
```

Existen varios programas que sirven para elaborar documentación técnica, entre ellos se tiene: *FrameMaker*, *Ventura Publisher*, *FreeHand*, *MS-Word*,...

Así, la instrucción `\software` cambiará el tipo de letra a *cursiva*. Con este tipo de instrucciones se puede crear la guía de estilos, ya que en caso de que se necesite modificar el formato, únicamente se tendrá que hacer en la definición de `\software`, en lugar de hacer el cambio en cada uno de los programas mencionados a lo largo del documento.

En este ejemplo también se observa cómo funciona la parte de *NúmArg*; *NúmArg* es un número entre 1 y 9 que indica cuántos argumentos obligatorios necesitará la nueva instrucción, y para utilizarlos se coloca el carácter # seguido del número del argumento correspondiente en la parte *Definición*.

La instrucción `\renewcommand*` sirve para modificar el comportamiento de una instrucción que ya existe. Por último, `\providecommand*` también define una instrucción, pero en caso de que la instrucción ya exista, la definición no tendrá efecto alguno. Todas estas instrucciones son locales, es decir, sólo serán válidas dentro del grupo donde sean definidas; por ejemplo, si se define una instrucción dentro de un entorno `center`, al terminar este entorno, dicha instrucción no existirá más.

Además de estas tres instrucciones existen sus respectivas versiones sin asterisco. La diferencia entre ambas versiones es la longitud del argumento, con asterisco no es posible utilizar las instrucciones de cambio de párrafo (`\par` o la línea en blanco) y sin asterisco sí es posible. La desventaja de utilizar las versiones con asterisco es la cantidad de memoria que utilizan, ya que primero se guarda en memoria el argumento o argumentos y luego se les aplica el formato.

Nuevos entornos

Cuando se necesita dar un formato especial a una gran cantidad de texto se tienen los entornos. Para definir un entorno se tienen las siguientes instrucciones:

`\newenvironment*{`*Nombre*`}[`*NúmArg*`]{`*DefInicio*`}{`*DefFinal*`}`
`\renewenvironment*{`*Nombre*`}[`*NúmArg*`]{`*DefInicio*`}{`*DefFinal*`}`

El entorno tendrá el *Nombre* que se defina y se utilizará como cualquier otro entorno (`\begin{`*Nombre*`}` y `\end{`*Nombre*`}`). A diferencia de las instrucciones, la diagonal (\) no se requiere para el *Nombre* del entorno.

En *DefInicio* se colocan todas las instrucciones que se ejecutarán cuando se escriba `\begin{`*Nombre*`}` y en *DefFinal* se colocan las que se ejecutarán al escribir `\end{`*Nombre*`}`. El siguiente ejemplo muestra cómo definir un nuevo entorno (tomado de Cascales Salinas et al. (2003, pág. 96)):

```
\newenvironment*{citas}[1]%
{\newcommand\Autor{#1}\begin{quote}\itshape}%
{\end{quote}\centerline{\Autor}}

\begin{citas}{Javier}
Mi carrera ha sido lenta como la del caracol,
pero segura y sólida como su concha.
\end{citas}
```

Mi carrera ha sido lenta como la del caracol, pero segura y sólida como su concha.
Javier

Como puede observarse, el entorno `citas` utiliza un entorno `quote`, además de que coloca el texto de la cita en cursiva.

Los argumentos en los entornos funcionan como en las instrucciones, en *NúmArg* se coloca un número del 1 al 9 para indicar el número de argumentos obligatorios y se utilizan con el carácter `#` seguido del número correspondiente. Pero es importante mencionar que los argumentos sólo pueden utilizarse en la parte *DefInicio*, así que para utilizarlos en la parte *DefFinal* se puede hacer lo mismo que en el ejemplo para colocar el nombre del autor de la cita.

La instrucción `\renewenvironment*` modifica el comportamiento de un entorno que ya existe. Y al igual que las instrucciones de la sección anterior, existen las versiones sin asterisco para definir o redefinir entornos y tienen las mismas restricciones; en estos casos es más común utilizar las versiones sin asterisco.

Contadores y longitudes

Además de instrucciones y entornos, LaTeX utiliza variables (como los demás compiladores) para componer el documento. Estas variables pueden ser de dos tipos: «contadores» o «longitudes». Un ejemplo de contador es el número de página y una longitud es el ancho del renglón.

Las instrucciones que manipulan un contador son:

`\setcounter{`*NombreContador*`}{`*Valor*`}`
`\addtocounter{`*NombreContador*`}{`*Valor*`}`

y para utilizar el valor del contador se tiene la siguiente sintaxis:

`\the`*NombreContador*

Por ejemplo:

```
\setcounter{chapter}{7}
Este es el capítulo~\thechapter, en la página%
~\thepage.
\addtocounter{chapter}{-1}
Ahora se disminuye el capítulo en uno:
\thechapter.
```

Este es el capítulo 7, en la página 72. Ahora se disminuye el capítulo en uno: 6.

De igual forma se tienen instrucciones para manipular longitudes, estas son:

`\setlength{`*NombreLongitud*`}{`*Valor*`}`
`\addtolength{`*NombreLongitud*`}{`*Valor*`}`

el *Valor* es un número real acompañado de su unidad de medida. Las unidades que LaTeX acepta se muestran en el cuadro 2.2 (página 2.2). Para mostrar el valor de la longitud se utiliza la siguiente sintaxis:

`\the`*NombreLongitud*

la longitud se expresa en puntos. Por ejemplo:

```
\setlength{\parindent}{1cm}
En este párrafo se modificó el sangrado
con la longitud \verb+\parindent+, y su
valor es de \the\parindent.
```

En este párrafo se modificó el sangrado con la longitud `\parindent`, y su valor es de 28.45274pt.

Otras instrucciones

Además de las instrucciones mencionadas en las secciones anteriores, es posible utilizar otras instrucciones propias de TeX. Las más comunes son las siguientes:

`\def{`*Nombre*`}#1..#9{`*Definición*`}`
`\let{`*Nombre*`}=\Instruccion`

el primero funciona como `\newcommand` y el segundo define la instrucción *Nombre* que hará lo mismo que la instrucción `\Instruccion` existente.

Por último, como (LA)TeX es un compilador, también proporciona instrucciones propias de los compiladores: condicionales y ciclos. Algunos condicionales son propios de TeX y otros de LaTeX y los ciclos son propios de TeX. No se explicarán en este trabajo, simplemente se mencionan ya que se utilizan con frecuencia para programar clases y paquetes. Para mayor referencia se puede consultar Cascales Salinas et al. (2003) y Lamport (1994) para LaTeX y Knuth (1986) para TeX, sólo se mostrará un ejemplo tomado de Cascales Salinas et al. (2003, pág. 483) con algunas modificaciones.

```
\newcount\minum
\newcount\param
\def\numeros#1{\ifnum#1<2 ninguno%
 \else 1\minum=1%
 \param=#1 \advance\param by -1%
 \loop \advance\minum by 1%
  \ifnum\minum<\param, \the\minum%
 \repeat%
\fi}
Los números enteros positivos menores que
20 son: \numeros{20};
y menores que 1: \numeros{1}.
```

Los números enteros positivos menores que 20 son: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19; y menores que 1: ninguno.

Herramientas para crear plantillas

Para crear plantillas, ya sean clases o paquetes, LaTeX cuenta con unas herramientas para facilitar esta tarea. El paquete doc permite desarrollar una plantilla en conjunto con su documentación en un único archivo. La ventaja de este método es que se podrá actualizar fácilmente la plantilla.

Una vez escrito este archivo, LaTeX junto con la herramienta DOCSTRIP se encargarán de generar por separado el archivo de la plantilla (con extensión `cls` o `sty`) y un documento `dvi` que contendrá la documentación de la nueva plantilla.

La idea de tener en un mismo archivo tanto el código como su documentación surgió cuando Donald E. Knuth creó TeX con ayuda del sistema WEB. De esta manera es fácil transportar TeX a cualquier plataforma.

En este trabajo se utilizarán estas herramientas para crear las plantillas.

3.1.2. Estructura de las plantillas

Todas las clases y paquetes de LaTeX tienen una estructura similar, de manera general esta estructura es la siguiente:

1. Identificación
2. Código inicial
3. Declaración de opciones
4. Ejecución de opciones
5. Inclusión de paquetes
6. Código principal

Todas estas partes son opcionales. La importancia de esta estructura es que permite tener un archivo organizado y fácil de entender.

A continuación se explican estas partes, además se tomará parte del código de la clase `book` como ejemplo.

Identificación

En la primera parte del archivo se coloca la información del mismo (si será una clase o un paquete), el nombre de la plantilla y de manera opcional se coloca la fecha de liberación («release») junto con una breve descripción de qué hace la plantilla.

Además se puede indicar cuál versión de LaTeX que necesita la plantilla para funcionar. Esta información es opcional.

La clase `book` comienza de la siguiente manera:

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesClass{book}[2004/02/16 v1.4f Standard LaTeX document class]
```

Indica en primer lugar que necesita la versión de LaTeX 2_ε del 1 de diciembre de 1995. La segunda línea indica que provee la clase llamada `book` y el comentario da la fecha en que fue liberada y una descripción de la clase.

Código inicial

En ocasiones se necesita declarar algunas variables o instrucciones antes de realizar cualquier acción, o se puede incluir algún paquete. Todo esto se coloca en esta parte.

La clase `book` define algunos condicionales para utilizarlos más adelante. Parte de este código es el siguiente:

```
\newcommand\@ptsize{}
\newif\if@titlepage
```

Aquí se define la instrucción `\@ptsize` que se utilizará para procesar la opción del tamaño de letra, y en la segunda línea se declara un condicional para utilizarlo en la opción *titlepage*

Declaración de opciones

Algunas plantillas requieren de opciones, como el caso del paquete babel que puede utilizar la opción *spanish*, o la clase book que puede colocar la tipografía en uno de tres tamaños. Todas las opciones que la plantilla podrá aceptar se declaran aquí, además de que pueden «pasarse» opciones a cierto paquete.

Algunas de las opciones que la clase book define son las siguientes:

```
\DeclareOption{letterpaper}
  {\setlength\paperheight {11in}%
  \setlength\paperwidth  {8.5in}}
\DeclareOption{10pt}{\renewcommand\@ptsize{0}}
```

Las opciones definidas corresponden, en primer lugar al tamaño del papel (carta en este caso) y luego al tamaño de la letra (10 puntos).

Ejecución de opciones

Una vez definidas las opciones, en esta parte se ejecutan, es decir, se define el código que se ejecutará al colocar cada opción.

La clase book realiza lo siguiente:

```
\ExecuteOptions{letterpaper,10pt,twoside,onecolumn,final,openright}
\ProcessOptions
```

En primer lugar se definen las opciones predeterminadas de la clase y luego se ejecutan.

Inclusión de paquetes

En ocasiones la plantilla requiere de algún otro paquete o clase ya definida. De esta forma se reduce la cantidad de código, facilitando la creación de la plantilla.

La clase book no carga ningún paquete.

Código principal

Por último, se define todo lo que hará la plantilla, puede definir instrucciones o entornos, dar un formato especial a ciertas partes del documento, etc.

En la clase book esta parte es muy amplia; una parte de este código es la siguiente:

```
\newcommand\section{\@startsection {section}{1}{\z@}%
  {-3.5ex \@plus -1ex \@minus -.2ex}%
  {2.3ex \@plus.2ex}%
  {\normalfont\Large\bfseries}}
```

Con esto se define la instrucción `\section`.

3.2. Plantillas para documentos técnicos

Lo que ahora corresponde es definir los elementos que contendrá cada una de las plantillas para los documentos técnicos descritos en la sección 1.3. En cada una de las siguientes secciones se diseñará la plantilla, mencionando la clase que se tomará como base, los paquetes que se utilizarán, las modificaciones que se harán y los nuevos elementos (instrucciones y entornos) de la plantilla.

La versión de LaTeX 2ε que se utilizará para desarrollar y probar las plantillas es la distribuida con MiKTeX versión 2.4.

3.2.1. Documentación de software

Como se mencionó en el primer capítulo, la documentación de software se forma por varios documentos pequeños, debido a que se elabora cuando menos uno para cada etapa del desarrollo del software y en ocasiones se requiere de dos versiones del mismo documento, uno para el equipo de desarrollo y otro para el cliente.

Clase base

Considerando las clases estándar de LaTeX (véase el cuadro 2.3 en la página 39) la clase más adecuada es `article`, debido a que fue diseñada para documentos cortos.

Se utilizará la versión 1.4f del 16 de Febrero del 2004 de la clase `article`.

Paquetes a utilizar

Existen dos paquetes que serán de utilidad para crear esta plantilla: `fancyhdr` y `titlesec`, con el primero se puede modificar la composición de la cabecera y pie de la página (véase la página 42) y con el segundo se modifica la presentación de las etiquetas para los capítulos y secciones mencionados en la sección 2.3.3.

Se utilizará la versión 3.0 del 3 de Marzo del 2004 del paquete `fancyhdr` y la versión 2.5 del 7 de Marzo del 2002 del paquete `titlesec`.

Nuevos elementos y modificaciones

Las principales modificaciones que se harán a la clase `article` son la composición de la portada, el formato de los títulos y cabeceras y los márgenes de la hoja. También se mostrará la bibliografía y el índice alfabético en la tabla de contenidos.

- La portada incluirá información del sistema (el nombre y la versión), el tipo de documento, los nombres de quienes elaboraron el documento y la fecha.
- Los principales cambios en la presentación de títulos son la tipografía que utilizará, además de que se modificará el comportamiento de la instrucción `\part` para facilitar la creación de un documento final de todo el sistema.
- En las cabeceras sólo se modificará la presentación, ya que las opciones de LaTeX las componen en mayúsculas, lo cual ocasiona que títulos largos no se muestren completos.

- Los márgenes se cambiarán, ya que LaTeX reserva un espacio para las notas al margen; estas notas no son muy utilizadas y ocasionan que se ocupen más páginas.

3.2.2. Manuales de usuario

Un manual de usuario, generalmente es un documento que consta de varios capítulos, en los cuales se presenta algún tema específico acerca del funcionamiento de algún aparato, sistema, etc. Estos documentos varían en su tamaño, desde 20 páginas hasta tomos de 300 páginas, por ejemplo.

Clase base

Para este caso, la clase más adecuada es `book`, ya que permite manejar capítulos y proporciona otras instrucciones útiles. No existe límite en la cantidad de páginas que puede contener un documento de esta clase.

Se utilizará la versión 1.4f del 16 de Febrero del 2004 de la clase `book`.

Paquetes a utilizar

De igual manera que en la documentación de software, se utilizarán los paquetes `fancyhdr` y `titlesec`.

Nuevos elementos y modificaciones

Las modificaciones se harán para esta plantilla en la portada, en la presentación de los títulos y cabeceras y en los márgenes, además de que se agregarán algunos entornos. También se mostrará la bibliografía y el índice alfabético en la tabla de contenidos.

- La portada incluirá el título del manual (como se mostró en el ejemplo de la página 23), la versión del sistema y el nombre de quienes lo elaboraron.
- El formato de los títulos se modificará de manera similar a la documentación de software.
- Las cabeceras se modificarán también como se mencionó en la documentación de software.
- En cuanto a los márgenes, se ocuparán 2/3 partes del ancho de la página.
- Se definirá el entorno para la sección «notas de la edición» (véase la figura 1.8).

3.2.3. Presentaciones

Para que una presentación logre su objetivo, ésta debe mostrar la información de manera clara, las diapositivas no deben contener mucho texto y los gráficos y sonidos únicamente deben aparecer cuando sea necesario.

Clase base

Para elaborar esta plantilla se tomará la clase `prosper` como base, ya que permite crear fácilmente un presentación para imprimirse o proyectarse, permite crear animaciones y no está limitada a un programa en especial para proyectarse correctamente.

Se utilizará la versión 1.5 del 17 de Julio del 2001 de la clase `prosper`.

Paquetes a utilizar

Los paquetes que necesita esta clase para funcionar son seminar, hyperref, graphicx y PSTricks, todos ellos son incluidos por la clase prosper. Además se utilizarán los paquetes mathpazo, amssymb y pst-grad para el texto de las diapositivas

Se utilizará la versión 1.4 del 13 de Octubre de 1997 de seminar, la versión 6.74m del 30 de Noviembre del 2003 del paquete hyperref, la versión 1.0f del 16 de Febrero de 1999 del paquete graphicx, la versión del 5 de Junio del 2004/05/12 del paquete ps-all (PSTricks), la versión 9.1b del 26 de Enero del 2004 del paquete mathpazo, la versión 2.2d del 22 de Enero del 2002 del paquete amssymb y la versión del 15 de Julio del 2004 del paquete pst-grad.

Nuevos elementos y modificaciones

La plantilla de las presentaciones se diseñará completamente, tomando los lineamientos de la clase prosper para crear estilos. El diseño se basará en los conceptos dados en la sección 1.3.3.

Los elementos que se diseñarán son: la presentación de la portada, el diseño del fondo para las diapositivas, los colores y tipografías para el texto de la presentación. Además, se incluirá una opción para crear una barra de navegación si se utilizará el programa ACROBAT de ADOBE para proyectar la presentación.

3.2.4. Escritos científicos

Como se mencionó en el primer capítulo, los escritos científicos se dividirán en dos: trabajos de investigación y trabajos de titulación. Se creará una clase para cada uno de estos trabajos.

Clase base

Para un trabajo de investigación se utilizarán las clases article y book, esto se debe a que un trabajo de investigación varía en su extensión y la clase article es adecuada para trabajos cortos, mientras que la clase book es mejor para un trabajo más extenso.

Para los trabajos de titulación se utilizará la clase book, ya que permite manejar varios capítulos.

Las versiones de estas dos clases serán las mismas que en las plantillas de la documentación de software y del manual de usuario.

Paquetes a utilizar

Se utilizarán los paquetes fancyhdr y titlesec, las mismas versiones que en la documentación de software. Además se utilizará el paquete graphicx para colocar el logotipo de la institución (específicamente el de la U.N.A.M., pero se podrá cambiar si se desea) en la portada.

Se utilizará la versión 1.0f del 16 de Febrero de 1999 del paquete graphicx.

Nuevos elementos y modificaciones

Las principales modificaciones que se harán para los trabajos de investigación son: la portada, los títulos y cabeceras, los márgenes y el resumen. Además se incluirá la bibliografía en la tabla de contenidos.

- La portada incluirá el nombre de la escuela, el nombre de la carrera, el nombre de la asignatura, el título del trabajo, el nombre de quienes lo elaboraron y la fecha.
- Los títulos y las cabeceras sufrirán las mismas modificaciones que en la documentación de software.
- En los márgenes se tendrán dos opciones, una que incluya el espacio para las notas al margen y la segunda que elimine este espacio.
- En la parte del resumen se incluirá una parte para las palabras clave.

Para los trabajos de titulación se harán modificaciones similares que en un trabajo de investigación: en la portada, los títulos y cabeceras, los márgenes y el resumen; además se crearán unos entornos.

- La portada incluirá el título que se obtendrá y el nombre del asesor.
- Los entornos que se crearán son para las secciones de agradecimientos, dedicatorias y epígrafe.

3.2.5. Convenciones

Para crear las plantillas se tomarán ciertas convenciones.

- Los nombres para cada plantilla serán los siguientes:

Nombre	Función
`aca-softdoc`	Documentación de software
`aca-manual`	Manual de usuario
`aca-slides`	Presentaciones
`aca-report`	Trabajos de investigación
`aca-thesis`	Trabajos de titulación

- Los nombres de las nuevas instrucciones serán en inglés, como las instrucciones de LaTeX: `\title`, `\author`, `\maketitle`, etc, asimismo para los entornos: `center`, `figure`, etc.
- La programación se hará utilizando la herramienta DOCSTRIP y el paquete `doc`.
- La documentación será en español.

Capítulo 4

Ejemplos de uso

Para terminar este trabajo, una vez programadas las plantillas mencionadas en el capítulo anterior, lo que corresponde es explicar cómo se podrán utilizar dichas plantillas. Se tomarán algunos trabajos elaborados a lo largo de la carrera como ejemplos de uso para cada plantilla; en el caso de la documentación de software y el manual de usuario, fueron tomados de Braude (2003) y de `http://www.imem.unavarra.es/3d_mec/spanish/index.php`, respectivamente.

Las plantillas que se crearon son las siguientes:

Nombre	Función
aca-softdoc	Documentación de software
aca-manual	Manual de usuario
aca-slides	Presentaciones
aca-report	Trabajos de investigación
aca-thesis	Trabajos de titulación

En este orden se explicará su funcionamiento. Principalmente se explicarán las opciones de la clase y los entornos e instrucciones que se hayan creado o modificado.

4.1. Documentación de software

Para elaborar la documentación de software se deberá utilizar la clase aca-softdoc. La estructura que se dio en la sección 1.3.1 se puede conseguir utilizando las instrucciones `\part`, `\section`, ..., `\subparagraph` mencionadas en el segundo capítulo.

4.1.1. Opciones

La clase aca-softdoc se deriva de la clase estándar article. La mayoría de las opciones de article están disponibles, excepto todas las relacionadas al tamaño y orientación del papel, únicamente se podrán crear documentos en tamaño carta.

Las opciones predeterminadas de esta nueva clase son: *letterpaper*, *10pt*, *oneside*, *onecolumn*, *final* y *notitlepage*.

4.1.2. Modificaciones

Como se mencionó en el capítulo anterior, a la clase article se le harían ciertas modificaciones, además de agregar algunos elementos. En esta sección se explicarán estos cambios.

Portada

Para crear la portada se cuenta con las instrucciones:

`\title{`*Nombre del sistema*`}`
`\subtitle{`*Tipo de documento*`}`
`\author{`*Autor(es)*`}`
`\date{`*Fecha*`}`

En `\title` se deberá colocar el nombre del sistema, mientras que `\subtitle` se creó para colocar el tipo de documento, por ejemplo: Plan de proyecto, Diseño técnico, etc. `\author` y `\date` se utilizan igual que en las clases estándar. Para crear la portada es necesario que el argumento de `\title` no sea vacío.

El diseño de la portada es diferente si se utilizó la opción *titlepage* o la opción *notitlepage*. Por ejemplo, si se introduce la siguiente información:

```
\title{Encuentro}
\subtitle{Plan de proyecto}
\author{Gaming Industries Consolidated}
\date{\shorttoday}
\maketitle
```

se puede obtener la portada de la figura 4.1(a), utilizando la opción *titlepage*, o la portada de la figura 4.1(b), utilizando la opción *notitlepage*.

Diseño de la página

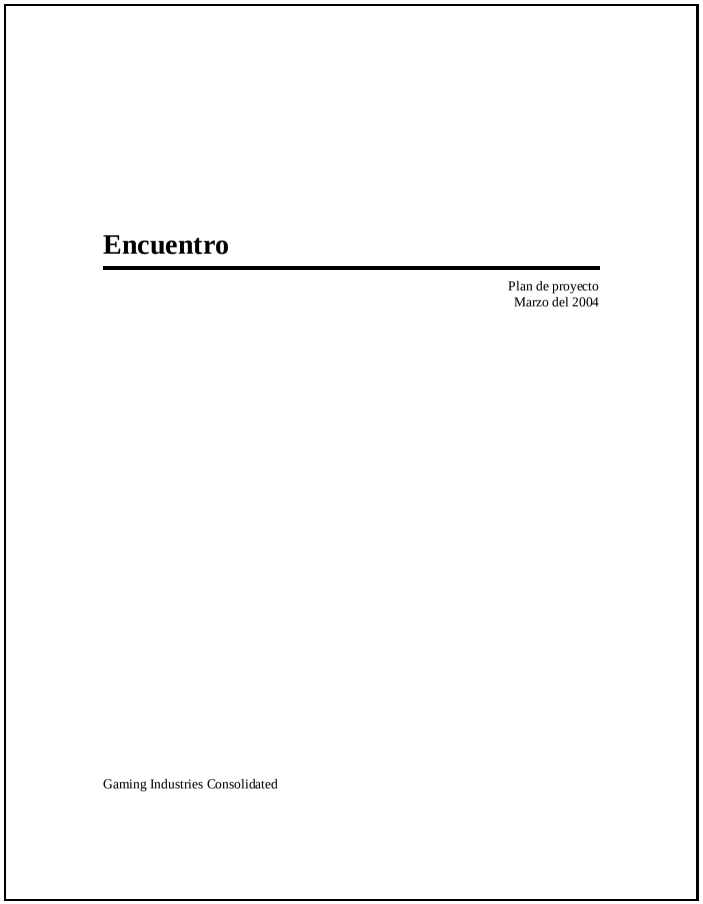
El diseño de las páginas fue modificado, con respecto a la clase estándar article, en lo siguiente:

- Los márgenes se fijaron en 1 pulgada.
- No será posible utilizar notas al margen, por lo que se eliminó el espacio que tenía destinado.
- Se redujo el tamaño de la tipografía para la cabecera y el texto ya no aparecerá en mayúsculas; además se agregó una línea debajo de la cabecera.
- En el pie aparece, alineado a la izquierda, el título y subtítulo del documento.
- La tipografía para los encabezados de sección se cambió a sans-serif y negrita.

En la figura 4.2 se pueden ver dos páginas donde se observan estos cambios.

Otras modificaciones

Resumen Es posible colocar el resumen del trabajo. Este apartado se colocará siempre centrado y a una sola columna (a diferencia de la clase estándar article donde el resumen se ve afectado por la opción *twocolumn*).

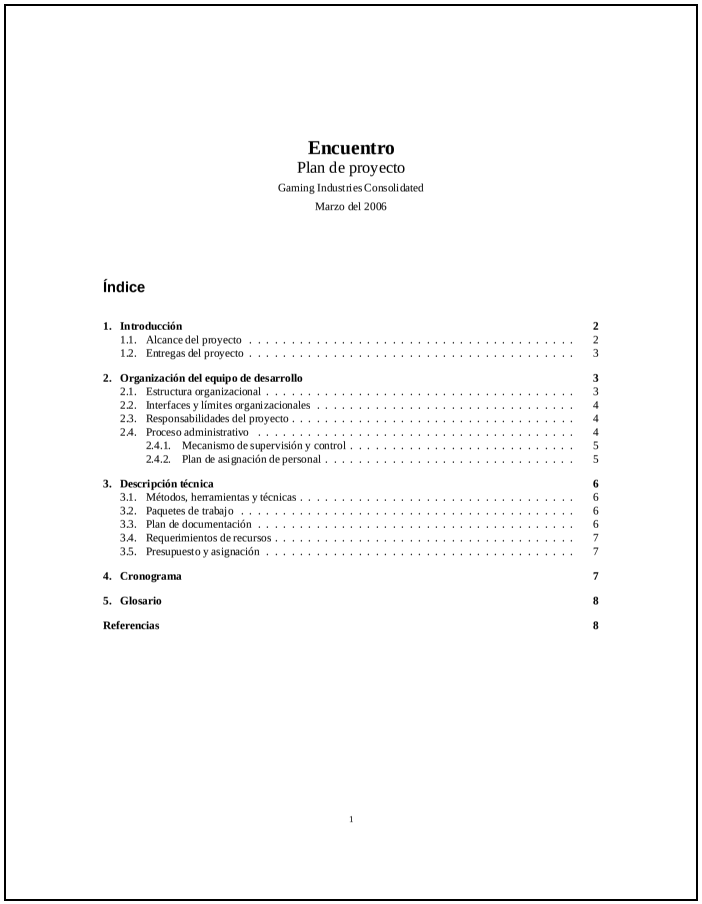


Encuentro

Plan de proyecto
Marzo del 2004

Gaming Industries Consolidated

(a) Opción *titlepage*



Encuentro
Plan de proyecto
Gaming Industries Consolidated
Marzo del 2006

Índice

1

(b) Opción *notitlepage*

Figura 4.1: Portadas generadas con la clase aca-softdoc

personaje extraño. Un reto del juego es que los nuevos valores de las cualidades tienen efecto sólo después de un retraso, que deja al jugador vulnerable durante cierto periodo.

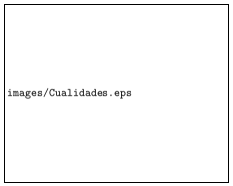


Figura 2: Uso de la intefaz para establecer valores de las cualidades

1.2. Entregas del proyecto

Lo siguiente deberá entregarse en los tiempos especificados:

- Versión 1 (prototipo) y documentación de apoyo enumerada: semana dos del mes dos.
- Versión 2 y documentación de apoyo enumerada: semana tres del mes cinco.

Los documentos de apoyo son: PPS, ERS, DDS, código fuente, código Java compilado, plan de mantenimiento de software y manual de usuario. (Las siglas se definen en la sección 5).

2. Organización del equipo de desarrollo

Las primeras dos versiones de este proyecto se ejecutarán según un proceso de desarrollo en espiral con una iteración correspondiente a cada versión. Las iteraciones deben agruparse de acuerdo con la clasificación usada en [USDP]. El USDP agrupa iteraciones en concepción, elaboración, construcción y transición. La primera iteración se considerará la única iteración de concepción. La segunda es la primera de las iteraciones de elaboración. Ésta será la versión 1 de Encuentro. El número de iteraciones subsecuentes y la naturaleza de la versión 2 debe decidirse después que el cliente ha visto una demostración.

2.1. Estructura organizacional

La figura 3 muestra la organización del proyecto Encuentro dentro de Gaming Industries. El proyecto debe organizarse como un equipo de colegas con papeles o roles designados. Los papeles son: líder del equipo, líder de administración de la configuración, líder de aseguramiento de calidad, líder de administración de requerimientos, líder de diseño y líder de implantación. Además, existen dos roles de enlace con comercialización y con el laboratorio de ingeniería de software. Estos roles se muestran en la figura 4. Cada miembro del equipo estará incluido en la inspección de todo el trabajo de los otros miembros, como se indica en la figura 4. Esto ocurrirá ya sea como parte de una inspección normal de grupo o, si el tiempo no le permite, una inspección realizada por el autor y su respaldo.

(a)

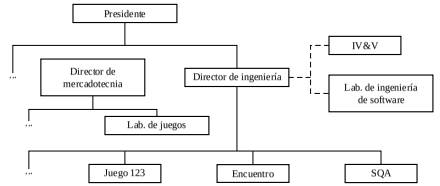


Figura 3: Gamming Industries Consolidates

Miembro	Líder de equipo	Líder de AC	Líder de QA	Líder de administración de requerimientos	Líder de diseño	Líder de implantación
Responsabilidad de enlace	Director de ingeniería			Mercadotecnia	Laboratorio ingeniería de software	
Responsabilidad de documento	PPS	PACS	PAQS	ERS	DDS	Código base

Cuadro 1: Responsabilidades del proyecto Encuentro

2.2. Interfaces y límites organizacionales

El equipo del proyecto debe interactuar con los siguientes individuos y organizaciones: director de ingeniería, comercialización, laboratorio de juegos, equipo V&V independiente y el laboratorio de ingeniería de software.

2.3. Responsabilidades del proyecto

Las responsabilidades de los participantes en proyecto se muestran en la cuadro 1. Ser responsable de un documento incluye lo siguiente:

- Asegurar que se cree el documento.
- Hacer que el líder del equipo identifique a los escritores del documento.
- Actualizar el documento durante todo el ciclo de vida del proyecto.

2.4. Proceso administrativo

La prioridad administrativa más alta debe ser el logro, de los parámetros de calidad especifica dos. La segunda prioridad es que el producto esté listo a tiempo. La tercera debe ser la satisfacción del mayor número

(b)

Figura 4.2: Ejemplo de páginas «normales» utilizando la clase aca-softdoc. En (a) se utilizó la opción *draft*

Modo borrador Cuando se utiliza la opción *draft*, además de realizar los cambios mencionados en el segundo capítulo, se modificó el pie: aparecerá la leyenda «THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW» (véase la figura 4.2(a)).

Formato de las leyendas La tipografía de las leyendas para los cuadros y las figuras se cambió: se redujo de tamaño y se colocará en tipografía sans-serif.

Nuevas instrucciones Se crearon las siguientes instrucciones:

\spacing Requiere de un argumento obligatorio, un número decimal. Con esta instrucción se modifica el interlineado del texto, por ejemplo, colocando como argumento `2`, el documento se compondrá a doble espacio. El valor predeterminado es `1`.

\shorttoday Coloca la fecha de compilación, pero únicamente el mes y el año.

\Section Funciona como `\section*`, coloca el encabezado de una sección, pero a diferencia de ésta, `\Section` agrega una entrada a la tabla de contenidos.

\includeBibliography Si se coloca esta instrucción en el preámbulo, se agregará la entrada de la bibliografía (o referencias) en la tabla de contenidos.

4.2. Manual de usuario

Para elaborar algún manual, la clase aca-manual es la más adecuada. De igual manera que en la documentación de software, la estructura del manual se consigue con las instrucciones `\part`, `\chapter`, ..., `\subparagraph`.

4.2.1. Opciones

La clase aca-manual tiene las mismas opciones que la clase book, excepto por la opción *twocolumn*, *notitlepage* y las que definen el tamaño de papel, las cuales fueron desactivadas. El tamaño de papel disponible es carta.

Las opciones predeterminadas de esta clase son: *letterpaper*, *10pt*, *twoside*, *onecolumn*, *final* y *openright*.

4.2.2. Modificaciones

A continuación se explican los cambios realizados a la clase book dentro de la nueva clase, aca-manual.

Portada

La creación de la portada utiliza, además de las instrucciones `\title`, `\author` y `\date` de las clases estándar, dos nuevas instrucciones:

`\subtitle{`*Subtítulo*`}`
`\logo{`*Archivo*`}`

La instrucción `\subtitle` sirve para colocar un subtítulo al manual. La instrucción `\logo` sirve para incluir algún logotipo y funciona igual que `\includegraphics`, es decir, el argumento obligatorio define la ruta y nombre del archivo gráfico y es posible modificarlo por medio del argumento obligatorio (el cual tiene la opción predeterminada de escalar el gráfico a una altura de 4.5 cm). El único argumento obligatorio para crear la portada es el de `\title`.

Por ejemplo, el siguiente código crea la portada de la figura 4.3(a):

```
\title{Manual de usuario de \MEC\ 1.9~beta}
\subtitle{Programa para el manejo numérico,\\
  simbólico y gráfico de expresiones vectoriales en mecánica}
\author{Javier Ros y José Ignacio Larraya\\
  Depto. Ingeniería Mecánica, Energética y de Materiales\\
  Universidad Pública de Navarra}
\date{Diciembre 1996}
\logo{3dmec}
\maketitle
```

Diseño de la página

El diseño de la página para esta clase es el siguiente:

- Los márgenes se fijaron en 1 pulgada para los lados superior, derecho e inferior de la página y en 2.3 pulgadas para el lado izquierdo
- La tipografía de los encabezados de sección será en sans-serif y negrita, además aparecerán recorridos 1.3 pulgadas a la izquierda con respecto al texto normal (este cambio no se realizó en la instrucción `\paragraph` ni en `\subparagraph`)

Además, se realizaron los mismos cambios en las notas al margen, cabecera y pie que en la clase aca-softdoc. En la figura 4.4 se pueden observar dos páginas como ejemplo de este diseño.

Otras modificaciones

Básicamente se realizaron las mismas modificaciones que en la clase aca-softdoc: al compilar con la opción *draft* aparecerá la leyenda «THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW» como se observa en la figura 4.4(a), y la tipografía de las leyendas se redujo de tamaño y se cambió a sans-serif.

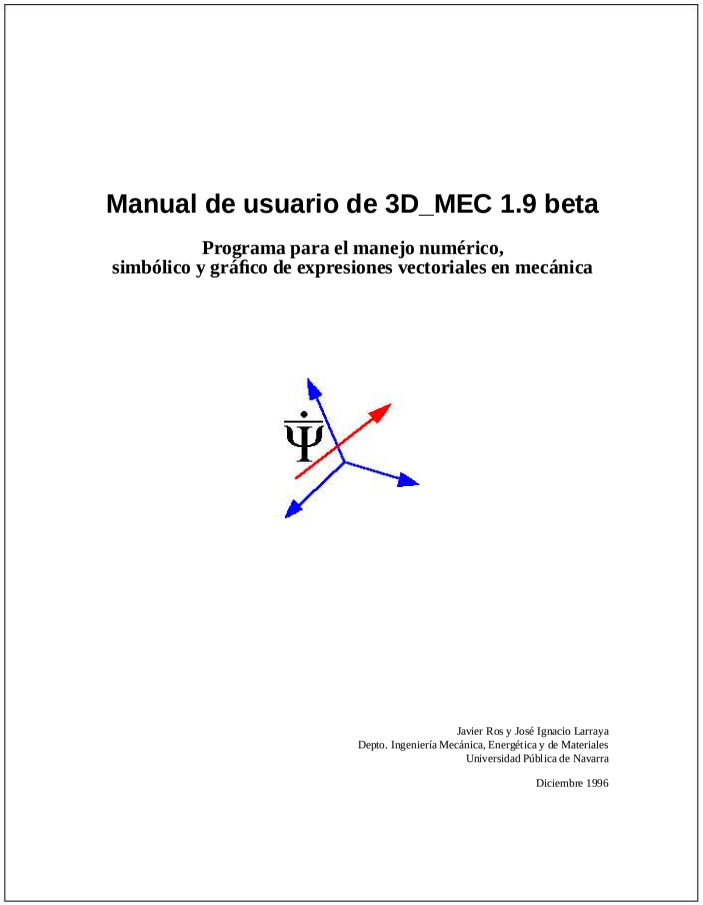
Nuevas instrucciones Se crearon las siguientes instrucciones:

`\spacing{`*interlinea*`}`
`\shorttoday`
`\includeBibliography`

las cuales funcionan igual que las creadas en la clase aca-softdoc. De igual manera, se crearon las instrucciones:

`\Chapter` Funciona de manera similar a `\Section`, descrita en la clase anterior, pero `\Chapter` sirve para los capítulos.

`\includeIndex` Incluye el índice alfabético en la tabla de contenidos, como lo hace la instrucción `\includeBibliography`.



(a) Portada

(b) Notas de la edición

Figura 4.3: Portada y notas de la edición generadas con la clase aca-manual

Figura 1.3: 3D_MEC después de ejecutar parte del *Fichero de Entrada*

1.2 Ventana principal

A continuación se describen algunos de los elementos del Menú que tienen relación directa con ésta ventana:

- **Archivo**
  - **Nuevo.** abre la con un *Fichero de Entrada* nuevo.
  - **Abrir.** abre la Ventana de Entrada con un *Fichero de Entrada* existente.
  - **Salir.** Abandona 3D_MEC.
- **Ventana**
  - **Borrar todo.** Elimina el contenido de la ventana activa.
  - **Cascada.** coloca las diferentes ventanas en forma de cascada.
  - **Mosaico Vertical.** Coloca las diferentes ventanas formando un mosaico vertical.
  - **Mosaico Horizontal.** Coloca las diferentes ventanas formando un mosaico horizontal.
- **Ejecutar**
  - **Play.** Hace que el *núcleo* comience la ejecución de las instrucciones del *Fichero de Entrada*.
  - **Continuar.** Hace que el *núcleo* continúe la ejecución de las instrucciones del *Fichero de Entrada* a partir del punto donde se detuvo.
  - **Pausa.** Detiene la ejecución de instrucciones. Se podrá reanudar la ejecución desde el punto donde se detuvo (**Continuar**).

THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW

(a)

Capítulo 2

El Núcleo de 3D_MEC

El *núcleo* es la parte de la aplicación que realiza los cálculos. El usuario utiliza el interface de 3D_MECpara interaccionar con éste.

En esta sección se describirán únicamente aquellos aspectos del *núcleo* que son necesarios, desde el punto de vista del Usuario, para trabajar con 3D_MEC.

De una forma muy simplista puede decirse que el *núcleo* ejecuta el conjunto de instrucciones contenidas en el *Fichero de Entrada*, y produce:

- Resultados que son dirigidos al *Fichero de Salida*, si son de tipo texto.
- Resultados que son dirigidos al *Dispositivo Gráfico* (Pantalla o Ventana, impresora...) si son de tipo gráfico.
- Errores o Avisos que son dirigidos al *Fichero de Salida de Errores*.

El *Fichero de Entrada*, el *Fichero de Salida*, el *Fichero de Salida de Errores* y el *Dispositivo Gráfico* tienen asociado su correspondiente elemento en el inteface de 3D_MECque permite su manipulación, edición, visualización, lectura, etc.. Deliberadamente se hace referencia a estos Ficheros y Dispositivos de una forma abstracta, ya que es así como son vistos o entendidos por el *núcleo* y no dependen de una versión o implementación particular del *interface* de 3D_MEC.

2.1 Estados del Núcleo

El *núcleo* puede encontrarse principalmente en tres estados:

- Detenido
- En pausa
- Procesando

2.1.1 Detenido

Puede decirse que en este estado el *núcleo* se encuentra recién Inicializado, o Reinicializado. Si el *núcleo* está en este estado la única acción que puede tomarse es iniciar la ejecución de instrucciones del *fichero de estado* a partir de la primera de ellas.

Este es el estado en que se encuentra el *núcleo* al arrancar la aplicación. También se puede llegar a este estado mientras se están ejecutando las Instrucciones:

(b)

Figura 4.4: Ejemplo de páginas «normales» utilizando la clase aca-manual. La página (a) se compiló con la opción *draft*

Nuevos entornos Además de estas instrucciones, se creó el entorno `edition`, el cual ayuda a crear las notas de la edición. Este entorno se deberá utilizar después de la instrucción `\maketitle` para que dichas notas aparezcan al reverso de la portada (figura 4.3(b)).

4.3. Presentaciones

Las presentaciones se pueden elaborar utilizando la clase aca-slides. La estructura de una presentación se puede ver en la figura 1.9 (página 27). Esta clase está basada en la clase prosper, mencionada en la sección 2.4, y por tanto, debe seguirse la estructura de dicha clase.

4.3.1. Opciones

Las opciones que ofrece aca-slides son las siguientes:

- *navibar*. Coloca en el pie de la diapositiva una barra de navegación con los siguientes botones:

Botón	Función
◁	Avanza una diapositiva
▷	Regresa una diapositiva
□	Opción de «pantalla completa»
×	Cierra el archivo

  Estos botones, sin embargo, funcionarán correctamente si se utiliza el programa ACROBAT.
- *print*. Si la presentación se imprimirá, se recomienda utilizar esta opción. La opción *print* pasa las opciones *slideBW*, *nocolorBG* y *accumulate* a la clase prosper. Como se observa, las diapositivas se compondrán en escala de grises y se deshabilitan las animaciones.
- *view*. Esta opción fue diseñada para crear una presentación que será proyectada. Al utilizarla se están pasando las opciones *slideColor*, *colorBG* y *noaccumulate* a prosper.
- *draft*, *final*, *total*, *nototal*, *ps* y *pdf*. Estas opciones funcionan exactamente igual que en la clase prosper.

La única opción predeterminada es *final*, las demás se deberán especificar en `\documentclass`.

4.3.2. Modificaciones

Debido a las especificaciones de prosper para crear estilos, se tuvo que crear el paquete PPRaca-slides el cual contiene el diseño de las diapositivas; no se recomienda utilizar este estilo directamente con prosper, ya que puede generar errores.

Portada

La portada se crea con la información de las siguientes instrucciones:

`\title{`*Título de la presentación*`}`
`\author{`*Autor(es)*`}`

`\subtitle{`*Subtítulo*`}`
`\email{`*E-mail(s)*`}`
`\institution{`*Institución*`}`
`\LogoTitle{`*Logotipo*`}`

Todas estas instrucciones funcionan como en la clase prosper, excepto `\LogoTitle`. La instrucción `\LogoTitle` incluye el archivo gráfico *Logotipo* en la portada; la altura de este archivo está dada por la longitud

`\LogoTitleSize`

cuyo valor predeterminado es de 2 cm. Por ejemplo, el siguiente código genera la portada de la figura 4.5:

```
\title{Distribución de terrenos}
\subtitle{Aplicación de Optimización no Lineal}
\author{Martínez Martínez Asael Fabian\and Ramírez Rodríguez Martha Elsa}
\email{fabian\_023@yahoo.com.mx\and martha@yahoo.com.mx}
\LogoTitle{logoUNAM}
```

Se creó una variante de la instrucción `\LogoTitle`, su versión con asterisco: `\LogoTitle*`. Esta nueva instrucción, en lugar de colocar un archivo gráfico, crea una «caja» cuadrada que permite colocar cualquier elemento de LaTeX. El tamaño de la caja también se define con `\LogoTitleSize`.

El valor predeterminado de la instrucción `\institution` es «U.N.A.M. – Facultad de Estudios Superiores Acatlán».

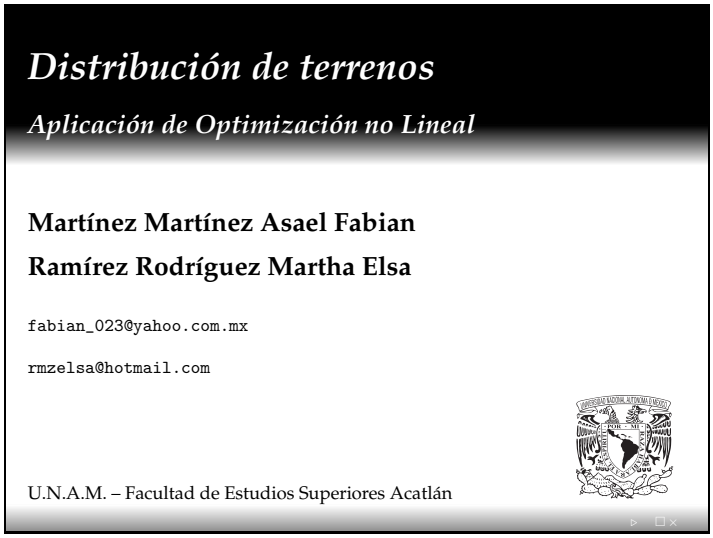


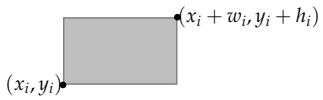
Figura 4.5: Portada generada con la clase aca-slides

Diseño de la diapositiva

En la figura 4.6 se pueden observar dos ejemplos de las diapositivas creadas con la clase aca-slides.

Objetivo

- Cada sección se puede representar con dos puntos:



- El objetivo es minimizar el perímetro de las cinco áreas:

$$\text{Min } f = 10 \sum_{i=1}^{5} 2\left(w_i + h_i\right)$$

logo

THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW

(a)

Resultados

Los resultados que arrojó el programa son los siguientes:

- La función objetivo es igual a `5.0859428827`
- Las variables toman los siguientes valores:

```
x3= 2.949942   h1= 1.724172   w1= 2.899942
x5= 4.656793   h2= 1.724172   w2= 2.899942
               h3= 3.017772   w3= 1.656851
y2= 1.774172   h4= 1.451655   w4= 3.444342
y4= 3.548344   h5= 4.267657   w5= 2.343206
```

- Regresando a las unidades originales, el costo de cercar las cinco secciones es de $ 50,859.43

Distribución de terrenos

(b)

Figura 4.6: Ejemplo de diapositivas generadas con la clase aca-slides. La diapositiva (a) se compiló con la opción *draft*

Al igual que una página, cada diapositiva se divide en tres partes: cabecera, cuerpo y pie. Estos tres elementos se explican a continuación.

Cabecera Como se mencionó en la sección 2.4, una diapositiva se crea con el entorno `slide` y el título de cada una se coloca como argumento en dicho entorno.

Cuerpo El contenido de la diapositiva se coloca dentro del entorno `slide`. Por ejemplo, la figura 4.6(b) se creó con el siguiente código:

```
\begin{slide}{Herramientas}
\begin{itemize}
\item Para resolver el problema se utilizó el programa \programa{Mathematica}, debido al número
  de restricciones
\item Se dividieron las cantidades entre 100, ya que el programa tardaba mucho
   en encontrar la solución
\item El modelo que se ingresó fue el siguiente:
  \begin{verbatim}
  0.2*(w1+h1+w2+h2+w3+h3+w4+h4+w5+h5)
  \end{verbatim}
\item Se utilizó la función: \verb+Minimize[f, restr, vars]+
\end{itemize}
\end{slide}
```

Es importante mencionar que todo lo que aparezca dentro del entorno `slide` se colocará dentro de una sola diapositiva, así que es importante revisar que aparezca todo el texto, ya que LaTeX no mandará ningún mensaje.

Dentro de una diapositiva es común colocar diagramas o imágenes. La clase prosper permite utilizar las herramientas PSTricks, que son un conjunto de paquetes que proporcionan varias instrucciones y entornos para crear imágenes vectoriales. Por ejemplo, la imagen de la figura 4.6(b) fue creada con el siguiente código:

```
\psset{unit=1cm, fillstyle=solid}
\psframe[linecolor=gray,fillcolor=lightgray](0,0)(2.2,1.3)
\psdots(0,0)(2.2,1.3)
\rput[r](0,0){\footnotesize$\left(x_i,y_i\right)$}
\rput[l](2.2,1.3){\footnotesize$\left(x_i+w_i,y_i+h_i\right)$}
```

Para mayor información sobre PSTricks se puede consultar Cascales Salinas et al. (2003). En la figura 4.7 se puede ver el espacio disponible para crear imágenes con PSTricks.

Logotipo Como se puede ver en las figuras anteriores, aparece un logotipo en la esquina inferior derecha. Este logotipo se puede crear de dos maneras:

1. Cuando se utiliza la instrucción `\LogoTitle`, el mismo archivo se utilizará para colocarlo en todas las diapositivas. En este caso el tamaño del logotipo será de 1 cm.
2. Utilizando la instrucción `\Logo` de prosper.

Si se utilizó la instrucción `\LogoTitle` pero se desea otro logotipo para las diapositivas se puede utilizar `\Logo` después de `\LogoTitle`; si no se desea colocar ningún logotipo se puede escribir `\Logo{}`.

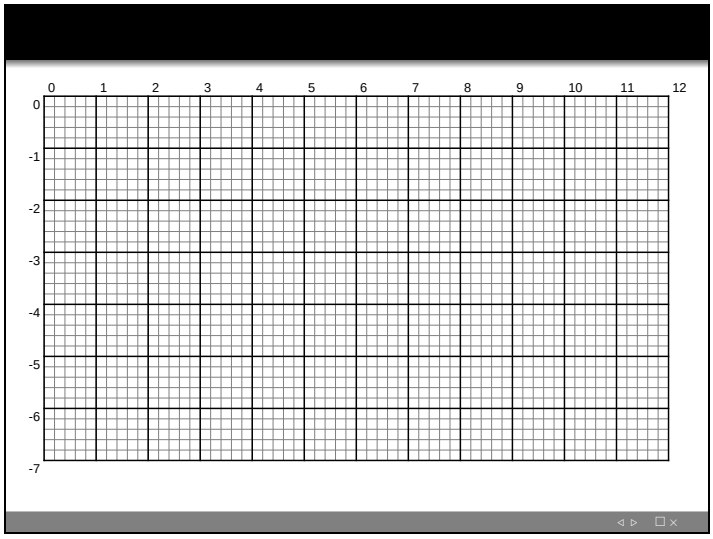


Figura 4.7: Espacio disponible para colocar imágenes, las unidades se dan en centímetros

Pie En la parte inferior de la diapositiva aparece, en la parte izquierda, la información colocada en la instrucción `\slideCaption`, o el título de la presentación, en caso de que no se haya utilizado esta instrucción.

Además, como se mencionó, la opción *navibar* colocará la barra de navegación. Si no se utilizó esta opción, en su lugar aparecerá el número de la diapositiva.

Otras modificaciones

Modo borrador Si se utiliza la opción *draft*, en el pie aparecerá la leyenda «THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW», además de realizar los cambios ya mencionados por esta opción.

Partes Cuando se utilice la instrucción `\part`, este encabezado aparecerá centrado en una diapositiva.

Referencias bibliográficas Cuando se utilice el entorno `thebibliography`, las referencias bibliográficas se colocarán en una diapositiva, así que no será necesario colocarlo dentro de un entorno `slide`. Los argumentos del entorno `thebibliography` son dos:

1. El argumento obligatorio es el mismo que en las clase estándar, define el ancho de la etiqueta.
2. El argumento optativo sirve para indicar el efecto de transición para esta diapositiva, como en el entorno `slide`.

Por ejemplo, el siguiente código genera la diapositiva de la figura 4.8

```
\begin{thebibliography}{XXXX}
  \bibitem[BV04]{} Boyd, Stephen y Lieven Vandenberghe.
  \textit{Convex Optimization}. Cambridge University Press. 2004.
\end{thebibliography}
```

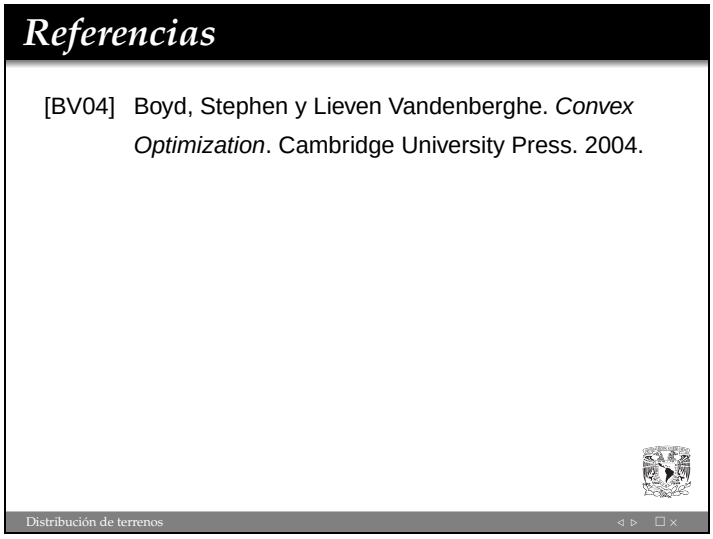


Figura 4.8: Ejemplo de las referencias bibliográficas creadas con la clase aca-slides

4.4. Trabajos de investigación

En el primer capítulo se dividieron los escritos científico-técnicos en dos: trabajos de investigación y trabajos de titulación. De la misma forma, se creó una clase para elaborar los trabajos de investigación y otra para los trabajos de titulación. En esta sección se explicará cómo utilizar la clase aca-report para crear trabajos de investigación.

4.4.1. Opciones

Un trabajo de investigación puede variar en su extensión, por tal razón la clase aca-report permite cargar una de dos clases: acticle o book. La clase article, como ya se ha mencionado, es más adecuada para trabajos cortos, mientras que la clase book es mejor si el documento se dividirá en capítulos (lo cual implica un trabajo más extenso). La forma de indicar qué clase se desea cargar es utilizando las opciones *article* o *book*. Es importante especificar alguna de estas dos opciones, de lo contrario, la compilación podría generar errores.

Opción *article*

Cuando se ha elegido la opción *article*, la clase aca-report aceptará las mismas opciones de la clase article.

Las opciones predeterminadas de la clase aca-report con la opción *article* son: *letterpaper*, *10pt*, *oneside*, *onecolumn* y *final*. La única opción que no se carga es la que define el tipo de portada (*titlepage* o *notitlepage*), por lo que deberá especificarse directamente.

Opción *book*

La opción *book* funciona de manera similar a *article*, pero se carga la clase book. Sin embargo, con la opción *book* se deshabilitan dos opciones de la clase estándar: *twocolumn* y *notitlepage*.

Las opciones predeterminadas de la clase aca-report con la opción *book* son: *letterpaper*, *11pt*, *twoside*, *onecolumn*, *final*, *openright* y *titlepage*.

Otras opciones

Independientemente de si se eligió *article* o *book*, las opciones que modifican el tamaño o la orientación del papel quedarán desactivadas y será posible utilizar una nueva opción: *notes*. La opción *notes* afecta a los márgenes de la página, más adelante se dará la explicación de esto.

4.4.2. Modificaciones

La clase aca-report realiza algunos cambios a la clase estándar que haya cargado, además de agregar nuevas instrucciones. Esto se explica a continuación.

Portada

Para crear la portada se tienen las siguientes instrucciones:

`\institution{`*Institución*`}`
`\faculty{`*Facultad o escuela*`}`
`\major{`*Carrera*`}`
`\subject[`*Grupo*`]{`*Asignatura*`}`
`\title{`*Título del trabajo*`}`
`\author{`*Autor(es)*`}`
`\date{`*Fecha*`}`

En la instrucción `\institution` se escribe el nombre de la escuela; el valor predeterminado para esta instrucción es: Universidad Nacional Autónoma de México. En `\faculty` se coloca la facultad donde se estudia, su valor predeterminado es: Facultad de Estudios Superiores Acatlán. El nombre de la carrera se especifica con `\major`. En `\subject` se especifica la materia y, como argumento optativo, el grupo. Por último, `\title`, `\author` y `\date` funcionan igual que en las clases estándar. Para crear la portada, únicamente se requiere que los argumentos de `\title`, `\author` y `\date` no estén vacíos, de lo contrario la clase aca-report mandará un error.

El diseño de la portada variará dependiendo de las opciones *article*, *book*, *titlepage* y *notitlepage*; en total se tienen tres diseños:

1. Utilizar las opciones *article* y *notitlepage*. Por ejemplo, ingresando los siguientes datos:

```
\subject[2802]{Sistemas Dinámicos II}
\title{Oscilador Armónico}
\author{Martínez Castañeda Luis Rubén \and
  Martínez Martínez Asael Fabian \\ Sosa Sánchez Diana}
\date{Junio del 2004}
\maketitle
```

se tiene la portada de la figura 4.9(a).

2. Tanto para *article* o *book* utilizar la opción *titlepage* y la instrucción `\maketitle`. Esta combinación creará una portada con el mismo diseño de la clase aca-thesis (veáse la siguiente sección).
3. Utilizar la combinación anterior, pero con la instrucción `\maketitle*`. En este caso, el diseño de la portada se puede ver en la figura 4.9(b); para este ejemplo se utilizó el siguiente código:

```
\subject[2802]{Series de tiempo I}
\title{El Saldo de la Deuda Externa Neta}
\author{Asael Fabian Martínez Martínez}
\date{Junio del 2004}
\maketitle*
```

Diseño de la página

El diseño de las páginas tiene las siguientes características:

- Los márgenes se fijaron en 1 pulgada, excepto si se utilizó la opción *notes*, la cual mantiene los márgenes de la clase estándar article o book, según sea el caso.
- El espacio para las notas al margen se mantendrá si se utilizó la opción *notes*, de lo contrario, se eliminará.
- Se redujo el tamaño de la tipografía para la cabecera y el texto ya no aparecerá en mayúsculas; además se agregó una línea debajo de la cabecera.
- En el pie quedará vacío, excepto en el inicio de capítulo (*book*), donde se mostrará el número de página.
- La tipografía para los encabezados de sección se cambió a sans-serif y negrita.

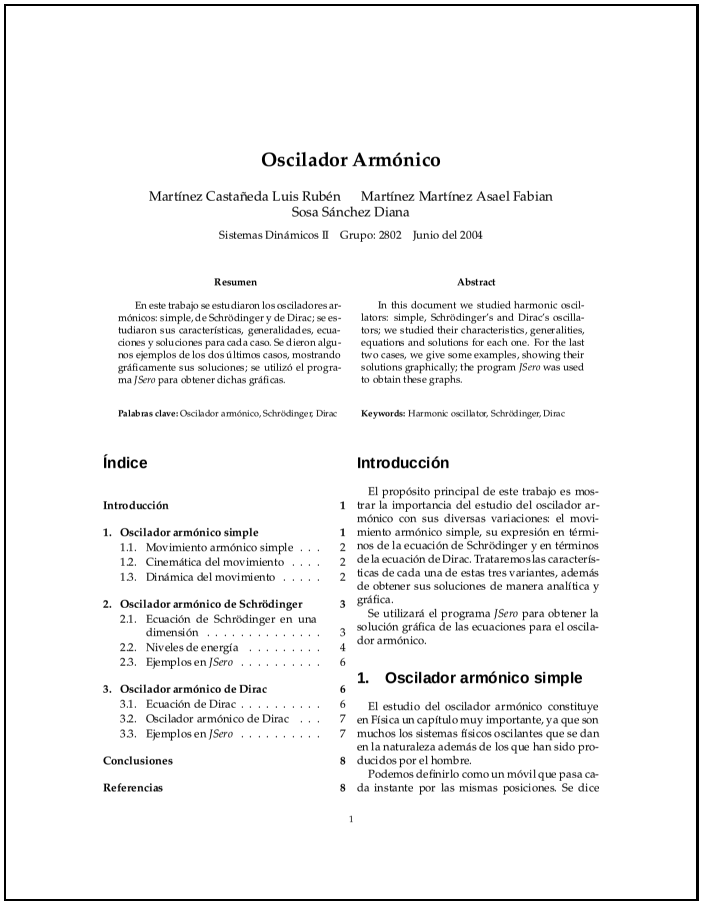
En la figura 4.10 se pueden ver dos páginas donde se observan estos cambios.

Otras modificaciones

Resumen Se hizo una modificación en el entorno `abstract` que manejan las clases estándar article y book. Dentro de la clase aca-report el entorno `abstract` tiene la siguiente estructura:

`\begin{abstract}{`*palabras-clave-español*`}{`*palabras-clave-inglés*`}`
Texto del resumen en español
`\englishAbstract`
Texto del resumen en inglés
`\end{abstract}`

El diseño del resumen es una tabla con dos columnas; la primera columna contendrá, de arriba hacia abajo: el encabezado **Resumen** centrado, el texto del resumen en español y las palabras clave en español del primer argumento; la segunda columna contendrá lo mismo que la primera, pero en inglés: el encabezado **Abstract** centrado, el texto del resumen en inglés y las palabras clave en inglés del segundo argumento. Como se puede observar, ambos resúmenes se colocan en un solo entorno



Oscilador Armónico

Martínez Castañeda Luis Rubén Martínez Martínez Asael Fabian
Sosa Sánchez Diana

Sistemas Dinámicos II Grupo: 2802 Junio del 2004

Resumen

En este trabajo se estudiaron los osciladores armónicos: simple, de Schrödinger y de Dirac; se estudiaron sus características, generalidades, ecuaciones y soluciones para cada caso. Se dieron algunos ejemplos de los dos últimos casos, mostrando gráficamente sus soluciones; se utilizó el programa *JSero* para obtener dichas gráficas.

Palabras clave: Oscilador armónico, Schrödinger, Dirac

Abstract

In this document we studied harmonic oscillators: simple, Schrödinger's and Dirac's oscillators; we studied their characteristics, generalities, equations and solutions for each one. For the last two cases, we give some examples, showing their solutions graphically; the program *JSero* was used to obtain these graphs.

Keywords: Harmonic oscillator, Schrödinger, Dirac

Índice

Introducción

El propósito principal de este trabajo es mostrar la importancia del estudio del oscilador armónico con sus diversas variaciones: el movimiento armónico simple, su expresión en términos de la ecuación de Schrödinger y en términos de la ecuación de Dirac. Trataremos las características de cada una de estas tres variantes, además de obtener sus soluciones de manera analítica y gráfica.

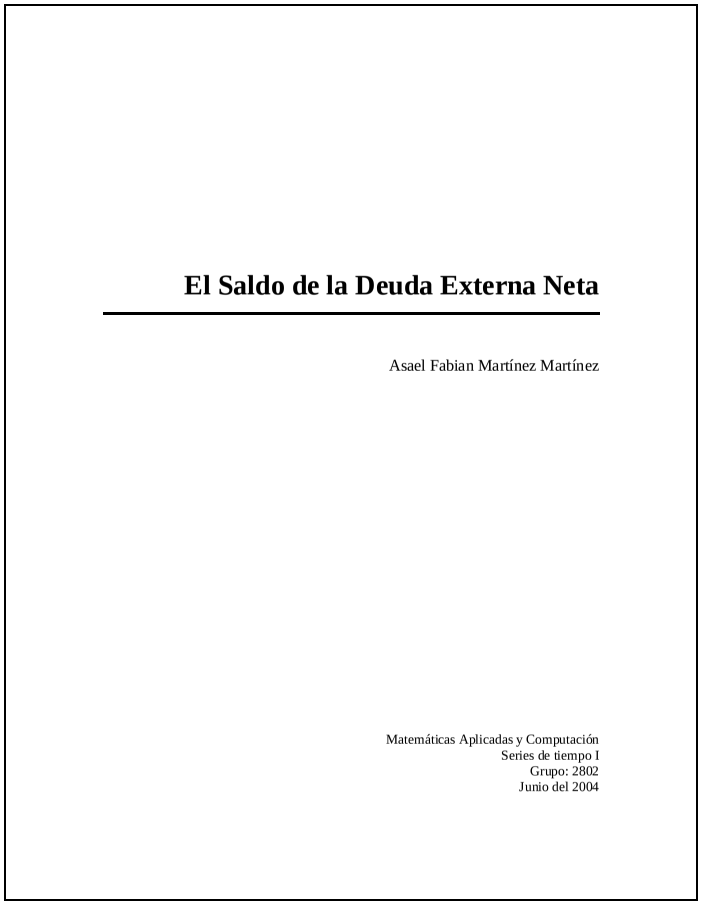
Se utilizará el programa *JSero* para obtener la solución gráfica de las ecuaciones para el oscilador armónico.

1. Oscilador armónico simple

El estudio del oscilador armónico constituye en Física un capítulo muy importante, ya que son muchos los sistemas físicos oscilantes que se dan en la naturaleza además de los que han sido producidos por el hombre.

Podemos definirlo como un móvil que pasa cada instante por las mismas posiciones. Se dice

(a) Opciones *article* y *notitlepage*



El Saldo de la Deuda Externa Neta

Asael Fabian Martínez Martínez

Matemáticas Aplicadas y Computación
Series de tiempo I
Grupo: 2802
Junio del 2004

(b) Opción *titlepage*, generada con `\maketitle*`

Figura 4.9: Portadas generadas con la clase aca-report

que el móvil ha efectuado una oscilación cuando se encuentra en la misma posición que la de partida y moviéndose en el mismo sentido.

1.1. Movimiento armónico simple

Una partícula describe un *movimiento armónico simple* cuando se mueve a lo largo del eje X; estando su posición x dada en función del tiempo t por la ecuación:

$$x = A\sin(\omega t + \varphi)$$

donde:

$$\begin{aligned} A &= \text{amplitud} \\ \omega &= \text{frecuencia angular} \\ \omega t + \varphi &= \text{la fase} \\ \varphi &= \text{fase inicial} \end{aligned}$$

Las características de un movimiento armónico simple son:

1. Como los valores máximo y mínimo de la función seno son $+1$ y -1, el movimiento se realiza en una región del eje X comprendida entre $+A$ y $-A$.
2. La función seno es periódica y se repite cada 2π, por tanto, el movimiento se repite cuando el argumento de la función seno se incrementa en 2π, es decir, cuando transcurre un tiempo $\omega(t+p)+\varphi = \omega t + \varphi + 2\pi$, con $p = \frac{2\pi}{\omega}$.

1.2. Cinemática del movimiento

En un movimiento rectilíneo, dada la posición de un móvil, obtenemos la velocidad derivando respecto del tiempo y, luego, la aceleración derivando la expresión de la velocidad.

La posición del móvil que describe un movimiento armónico simple en función del tiempo viene dada por la ecuación:

$$x = A\sin(\omega t + \varphi)$$

Derivando con respecto al tiempo, obtenemos la velocidad del móvil, denotada por la letra v, la cual queda como sigue:

$$v = A\omega\cos(\omega t + \varphi)$$

Derivando de nuevo respecto del tiempo, obtenemos la aceleración del móvil, la cual se denota con la letra a y está dada por la siguiente expresión:

$$a = -A\omega^2\sin(\omega t + \varphi)$$

La cual se puede simplificar quedando de la siguiente manera:

$$a = -A\omega^2 x$$

1.3. Dinámica del movimiento

La segunda ley de Newton nos da la fuerza necesaria para que un móvil de masa m describa un movimiento armónico simple. Esta fuerza es proporcional al desplazamiento x y de sentido contrario a éste

$$F = ma$$

Pero sabemos que la aceleración de un movimiento armónico simple, esta dada por $a = -A\omega^2 x$ de aquí que la ecuación de la fuerza quede de la siguiente manera:

$$F = -m\omega^2 x$$

Dado que la A es una constante y la masa también, se tiene que $-A\,(m) = -m$ en la ecuación. Dicha fuerza es conservativa y la energía potencial E_p correspondiente se halla integrando

$$F = -\frac{dE_p}{dx}$$

donde $E_p = \frac{1}{2}m\omega^2x^2$.

Se ha tomado como nivel cero de la energía potencial $E_p = 0$ cuando el móvil está en el origen, $x = 0$.

La energía total E, es la suma de la energía cinética E_k y de la energía potencial E_p. Se puede verificar que la energía total es constante e igual a:

$$\begin{aligned} E &= E_k + E_p \\ &= \frac{1}{2}mv^2 + \frac{1}{2}m\omega^2x^2 \\ &= \frac{1}{2}m(v^2 + \omega^2x^2) \end{aligned}$$

(a) Opción *article*

Capítulo 3

¿Cuál es el mejor modelo?

Ahora nos corresponde entrar en la etapa de estimación de los parámetros y diagnóstico para nuestros modelos tentativos. Recordemos que tenemos los siguientes modelos:

$$Y_t = \delta + \phi_1 Y_{t-1} + e_t \tag{3.1}$$

y

$$Z_t = \delta + e_t \tag{3.2}$$

los cuales son $AR(1)$ y ruido blanco, respectivamente. Aumentamos una variable δ porque suponemos que nuestro modelo tiene alguna constante.

3.1. Estimación

Tendremos que verificar que nuestros modelos no tengan estacionaridad ni estén sobreespecificados o subespecificados. Para que nuestro modelo sea factible sus residuales deben comportarse como ruido blanco.

3.1.1. Modelo $\mathbf{AR(1)}$

Para este modelo obtenemos los siguientes resultados para los parámetros:

Parámetro	Estimación	P-value
ϕ_1	0.941169	0.000000
media	82573.1	0.000000
δ	4857.89	—

Tomando $\alpha = 0.05$ podemos decir que el parámetro es necesario para el modelo si P-value$< \alpha$. Entonces, el parámetro ϕ_1 sí debe ir en nuestro modelo. De igual forma la constante δ es necesaria para el modelo, esto se determina con el P-value de la media

Entonces, nuestro modelo 3.1 queda de la siguiente manera:

$$Y_t = 4857.89 + 0.94Y_{t-1} + e_t \tag{3.3}$$

8

(b) Opción *book*

Figura 4.10: Ejemplo de páginas «normales» utilizando la clase aca-report

`abstract`, el resumen en español deberá ir primero y para indicar que se comienza el resumen en inglés se deberá utilizar la instrucción `\englishAbstract`.

Un ejemplo del funcionamiento de este entorno se puede observar en la figura 4.9(a); el siguiente fragmento de código generó el resumen:

```
\begin{abstract}{Oscilador armónico, Schrödinger, Dirac}{Harmonic oscillator, Schrödinger, Dirac}
En este trabajo se estudiaron los osciladores armónicos: simple, de Schrödinger y de Dirac; ...
\englishAbstract
In this document we studied harmonic oscillators: simple, Schrödinger's and Dirac's oscillators; ...
\end{abstract}
```

La información del entorno `abstract` siempre aparecerá centrada y no se verá afectada por la opción *twocolumn* como en las clases estándar.

Modo borrador Cuando se utiliza la opción *draft*, además de realizar los cambios mencionados en el segundo capítulo, se modificó el pie: aparecerá la leyenda «THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW».

Formato de las leyendas La tipografía de las leyendas para los cuadros y las figuras se cambió: se redujo de tamaño y se colocará en tipografía sans-serif.

Nuevas instrucciones En la clase `aca-report` se crearon las siguientes instrucciones:

`\spacing{`*interlínea*`}`
`\shorttoday`
`\Section{`*título de sección*`}`
`\Chapter{`*título de capítulo*`}`
`\includeBibliography`

La instrucción `\spacing` funciona igual que en la clase `aca-softdoc`, modifica la interlínea del documento. `\shorttoday` colocará la fecha de compilación, únicamente mes y año. `\includeBibliography` se coloca en el preámbulo, y su función es incluir el encabezado de la bibliografía (o referencias) en la tabla de contenidos.

La instrucción `\Section` funciona como `\section*`, pero el *título de sección* aparecerá en la tabla de contenidos; esta instrucción es útil para colocar, por ejemplo, el encabezado de la introducción del trabajo o las conclusiones, los cuales no deben ir numerados. La instrucción `\Chapter` funciona de manera similar a `\Section`, pero `\Chapter` crea encabezados de capítulos. La instrucción `\Section` estará disponible si se utilizó la opción *article*, mientras que `\Chapter` estará disponible utilizando la opción *book*.

4.5. Trabajos de titulación

La última de las clases elaboradas para crear documentos técnicos es `aca-thesis`. Con esta clase se pueden elaborar los trabajos de titulación.

4.5.1. Opciones

La clase aca-thesis se deriva de la clase estándar book y, por tanto, será posible utilizar las mismas opciones de esta última clase a excepción de las opciones *notitlepage*, *twocolumn* y aquellas que modifican el tamaño y la orientación del papel.

La clase aca-thesis carga siguientes opciones de forma predeterminada: *letterpaper*, *11pt*, *twoside*, *onecolumn*, *final* y *openright*.

4.5.2. Modificaciones

Los cambios que se realizan a la clase book son básicamente los mismos que se han venido realizando en las demás clases. A continuación se explican estos cambios.

Portada

La portada se crea con las siguientes instrucciones:

```
\institution{Institución}
\faculty{Facultad o Escuela}
\degree{Grado}{Carrera}
\title{Tipo-trabajo}{Título}
\author{Autor(es)}
\date{Lugar}{Fecha}
\advisor{Grado-asesor}{Nombre del asesor}
\logopath
```

En la instrucción \institution se escribe el nombre de la institución. En \faculty se coloca la facultad o escuela donde se estudió. Para indicar el grado que se obtendrá se utiliza el primer argumento de \degree y en el segundo argumento se coloca la carrera. La instrucción \title necesitará de dos argumentos: el tipo de trabajo (Tesis, Tesina, etc.) y el título del mismo. La instrucción \author funciona igual que en la clase book. La instrucción \date necesita dos argumentos: el lugar y la fecha. En la instrucción \advisor se colocará, en el primer argumento, el grado del asesor y en el segundo, su nombre. Ninguna de estas instrucciones debe omitirse para generar la portada, de lo contrario habrá errores.

Para simplificar la creación de la portada, la clase aca-thesis tiene las siguientes opciones predeterminadas:

```
\institution{Universidad Nacional Aut\'onoma de M\'exico}
\faculty{Facultad de Estudios Superiores Acatl\'an}
\date{Naucalpan, Estado de M\'exico}{\shorttoday}
```

La instrucción \logopath contiene la ruta y el nombre del archivo gráfico del logotipo que se incluirá en la portada, su valor predeterminado es images/logoUNAM. Para modificar esta ruta, simplemente se redefine esta instrucción.

En ocasiones al asesor se le conoce con otro nombre, por ejemplo, director; para modificar este título se tiene la instrucción \advisorname, la cual tiene como valor predeterminado «Asesor».

Por ejemplo, la portada de la presente tesis fue elaborada con el siguiente código:

```
\degree{Licenciado}{\Mac}
\title{Tesis}{Programación de plantillas\\
  para documentos técnicos utilizando \LaTeX}
\author{Asael Fabian Martínez Martínez}
\advisor{Mtra.}{MariCarmen González Videgaray}
\renewcommand*\advisorname{Asesora}
\maketitle
```

Diseño de la página

El diseño de las páginas tiene las siguientes características:

- Los márgenes se fijaron en 1.2 pulgadas; se tomó un margen mayor que en las demás clases, ya que se consideró el encuadernado del documento, donde generalmente se necesita reducir el tamaño de las hojas.
- Se eliminó el espacio destinado a las notas al margen, y de igual manera, la instrucción para generarlas.
- Se redujo el tamaño de la tipografía para la cabecera y el texto ya no aparecerá en mayúsculas; además se agregó una línea debajo de la cabecera.
- El pie quedará vacío, excepto en el inicio de capítulo, donde se mostrará el número de página.
- La tipografía para los encabezados de sección se cambió a sans-serif y negrita.

Este diseño se puede ver aplicado en este documento.

Otras modificaciones

Resumen El entorno `abstract` funciona igual que en la clase aca-thesis: se tendrá que escribir el resumen y las palabras clave en español y en inglés.

Modo borrador Cuando se utiliza la opción *draft*, además de realizar los cambios mencionados en el segundo capítulo, se modificó el pie: aparecerá la leyenda «THIS DOCUMENT HAS BEEN COMPILED IN DRAFT MODE, GRAPHICS WILL NOT SHOW».

Formato de las leyendas La tipografía de las leyendas para los cuadros y las figuras se cambió: se redujo de tamaño y se colocará en tipografía sans-serif.

Nuevas instrucciones Se crearon las siguientes instrucciones:

`\spacing{`*interlínea*`}`
`\shorttoday`
`\Chapter{`*título de capítulo*`}`
`\includeBibliography`

las cuales funcionan como en la clase aca-report (véase la página 99).

Nuevos entornos Por último, se crearon tres entornos para crear las secciones que aparecen en el preámbulo de la figura 1.11: `thank`, `dedication` y `epigraph`.

- El entorno `thank` crea la sección de los agradecimientos; el encabezado «Agradecimientos» está guardado en la instrucción `\thankname`.
- La sección de dedicatorias se crea con el entorno `dedication`; este entorno alineará a la derecha el texto.
- El entorno `epigraph` coloca el epígrafe al inicio del documento; este entorno necesita un argumento obligatorio, el cual es el autor del epígrafe, que se colocará al final del mismo.

En general, en todo este documento se puede observar el uso de la clase `aca-thesis` con sus instrucciones y entornos.

Más información

En el CD que se anexa a este trabajo (véase el ápendice B) se puede consultar la documentación de cada una de las plantillas. Esta documentación incluye el código completo de la plantilla junto con una explicación del mismo. Asimismo, se puede revisar cada uno de los documentos que se utilizaron como ejemplo, tanto el código fuente como el documento final.

Conclusiones

La información siempre ha sido muy importante para toda la humanidad, de tal forma que la aparición de escritos hizo una división en el estudio de la Historia. Actualmente, gracias a la tecnología, una persona puede acceder a cualquier tipo de información de cualquier parte del mundo fácilmente; por esta razón fue necesario saber cómo organizar y presentar la información de manera apropiada, surgiendo así la documentación técnica.

Con ayuda de la documentación técnica, además de que los documentos se presentan de manera clara y atractiva al lector, rompen las barreras de lenguaje que cada disciplina tiene. Por ejemplo, un biólogo difícilmente estaría interesado en las propiedades de la transformada de Hilbert, pero sí tendría gran interés en conocer algún programa de computadora que le permita estudiar las imágenes tomadas con un microscopio; en este caso, la documentación técnica ayuda a que el biólogo conozca, a través de artículos o manuales (elaborados por un egresado de M. A. C., por ejemplo), cuál es el mejor programa para su estudio.

En este trabajo se han dado los lineamientos para que un documento sea efectivo, es decir, que la información que contiene pueda ser asimilada fácilmente por el lector; para lograrlo, es necesario que el contenido del documento sea objetivo, preciso, claro y conciso. Asimismo se explicó la metodología que se debe seguir para elaborar documentos técnicos. Como se pudo observar, esta metodología es semejante a la seguida para desarrollar sistemas de cómputo. Por último, se explicó en dónde se aplica la documentación técnica en la carrera de Matemáticas Aplicadas y Computación: documentación de software, manuales de usuario, presentaciones y trabajos de investigación; estos documentos frecuentemente deben ser desarrollados por los estudiantes y egresados de esta carrera, por eso es necesario conocer cómo elaborarlos correctamente.

Un elemento muy importante dentro de la documentación técnica, y que es utilizado dentro de nuestra carrera, es la computadora. Como se mencionó anteriormente, existen varios programas de cómputo creados para elaborar documentos técnicos, los cuales facilitan todo el proceso que sigue un documento. En este trabajo se utilizó LaTeX 2_ε, para elaborar los documentos ya mencionados. Se eligió este lenguaje debido a que es el más adecuado para un estudiante de M. A. C., principalmente porque la formación recibida en esta carrera da las bases necesarias para aprender distintos lenguajes de programación, además, la sintaxis es similar a los lenguajes de programación para páginas web. Quienes utilizan LaTeX se pueden olvidar de cuestiones tipográficas y de diseño y los documentos obtenidos son de buena calidad. Además, es importante mencionar que LaTeX, no está limitado a una sola plataforma, es posible elaborar un documento en LINUX y revisarlo sin problemas en WINDOWS o MAC OS, por ejemplo; añadiendo a esto que el tamaño de los archivos fuente es insignificante en comparación con un documento con formato (uno elaborado en MS-WORD, por ejemplo).

En cuanto a las plantillas, que es el objetivo principal de este trabajo, con la elaboración de las cinco clases para LaTeX 2ε, se facilita la creación de documentos técnicos. Se pretendió hacer un trabajo que fuera útil para quienes elaboran documentos técnicos con LaTeX, es decir, los alumnos de M. A. C. en primer lugar, pero en general para cualquiera que utilice LaTeX.

LaTeX posee una gran potencia por ser un lenguaje de programación, ayudando así a que el escritor simplifique algunas de sus tareas; además, las clases y paquetes disponibles ayudan a elaborar diseños más complicados. Y si LaTeX no es suficiente, es posible utilizar TeX, que maneja conceptos de más bajo nivel, permitiendo manipular carácter por carácter.

Es importante que los estudiantes de M. A. C. conozcan lo que es la documentación técnica y cómo auxiliarse de la computadora para crear los documentos, debido a que frecuentemente tendrán que relacionarse con distintas disciplinas para elaborar cualquier trabajo, y la comunicación que exista debe ser clara y precisa (ya sea de manera verbal o por escrito). Además, haciendo una analogía con la alfabetización, considero que los conocimientos recibidos en esta carrera nos ayudan a «leer» y comprender el lenguaje de las matemáticas aplicadas y de la computación, pero la documentación técnica nos ayuda a «escribir» matemáticas aplicadas y computación en español.

Este trabajo da una introducción al lenguaje LaTeX, pero lo que ahora corresponde hacer es que LaTeX se utilice con más frecuencia en los estudiantes de Matemáticas Aplicadas y Computación. Actualmente (Febrero del 2006), hay un proyecto en el Centro de Desarrollo Tecnológico de la Facultad de Estudios Superiores Acatlán que pretende poner en línea un curso de LaTeX 2ε; sin embargo, es necesario que dentro de la Facultad también se conozca este trabajo.

Como parte de mi servicio social y apoyo al Centro de Desarrollo Tecnológico, he impartido algunos cursos de LaTeX 2ε a alumnos y profesores, a quienes les ha parecido una herramienta útil para escribir sus trabajos. Gracias a esto se ha comenzado con la difusión de LaTeX y ya existen algunos trabajos de M. A. C. y Actuaría creados con este lenguaje.

Alguna vez un profesor me comentó que los estudiantes de Matemáticas Aplicadas y Computación no se dedicarán a escribir, y por tanto un trabajo sobre LaTeX no tiene importancia alguna. Pero precisamente porque estas personas no tienen idea alguna de lo que es escribir (aunque la documentación técnica los educa en este aspecto) y todo lo que ello involucra (cuestiones de diseño, tipografía, tamaños de papel, etc.) es necesario que conozcan y utilicen LaTeX para crear sus documentos. De esta manera, ellos pueden dedicarse a difundir lo que saben: Matemáticas Aplicadas y Computación (que al fin y al cabo implica escribir) y le dejan a LaTeX lo que no saben (el diseño y todo lo demás).

Además, este trabajo es útil para los estudiantes de M. A. C. (aunque también puede serlo para otras carreras) ya que actualmente no existe un formato «oficial», o cuando menos estándar, para elaborar trabajos de titulación. De igual forma, a los estudiantes no se les explica cómo elaborar un manual de usuario, por ejemplo, a pesar de que cursan la materia de Ingeniería de Software (y como se explicó en este trabajo, un manual es necesario para entrenar al cliente en el uso del sistema); y frecuentemente se piden, a lo largo de la carrera, trabajos de investigación y presentaciones, sin embargo, no se dan los elementos que deben contener estos documentos.

Otra aportación útil de este trabajo es que recientemente se estableció como requisito en la Biblioteca Central, entregar un ejemplar del trabajo de titulación en formato `pdf`. Como se mencionó, LaTeX maneja archivos `pdf` sin problema alguno, algo que programas como MS-WORD no hacen, y a pesar de que existe software adicional que convierte archivos `doc` en `pdf`, es difícil conseguirlo o

utilizarlo. Además, dicho ejemplar electrónico debe estar fragmentado por capítulos[1], esto tampoco es problema si se trabaja con LaTeX, ya que hay instrucciones que permiten hacerlo fácilmente.

Extensiones de este trabajo

Por otra parte, algunos temas relacionados a (La)TeX que son interesantes para su estudio son los siguientes:

- Lenguajes de marcado
- Programación literaria
- Software libre
- Diseño gráfico

A continuación se da una breve explicación de estos temas.

Lenguajes de marcado

Hay una gran variedad de lenguajes derivados de SGML, entre ellos HTML, MathML, XML y DocBook. XML y DocBook son utilizados frecuentemente para crear documentos técnicos, sin embargo esta información se presenta en formato electrónico. En este caso, sería útil crear un programa que pueda convertir un documento en XML o en DocBook a LaTeX para obtener una versión impresa del documento.

Actualmente, la ayuda de muchos del los programas de cómputo viene en un formato HTML, pero modificado para presentarse con el programa HTML Help. Un trabajo que podría hacerse en este caso es crear algún programa que convierta documentos LaTeX a ese formato.

Programación literaria

La programación literaria fue creada por Donald E. Knuth cuando programó TeX. Como se mencionó, el propósito de la programación literaria es tener en un mismo archivo el código del programa y su respectiva documentación, luego, con ayuda de algunas herramientas, se obtiene por una parte el código y por otra la documentación, en archivos independientes. El sistema WEB sirve para estos propósitos, el cual tiene varias versiones para diferentes lenguajes de programación y la documentación se genera en lenguaje TeX o LaTeX.

La investigación sobre este tema puede ser de utilidad, principalmente con fines de enseñanza. Se podría utilizar algún sistema de programación literaria, por ejemplo CWEB (que genera código en lenguaje C), para facilitar la enseñanza de programación estructurada en la carrera de M. A. C. Además, con el uso de la programación literaria se podría disminuir el hábito de no documentar el software.

Los principios de la programación literaria se han aplicado en los lenguajes .NET. Estos lenguajes utilizan XML para generar la documentación contenida en el código.

[1] Para más referencia sobre este requisito se puede consultar `http://bc.unam.mx/`.

Software libre

El software libre puede ser una herramienta muy útil en la enseñanza. En internet es posible encontrar una gran variedad de software libre creado para distintos propósitos, y, en la mayoría de los casos, el software viene bien documentado, lo cual ayudaría en gran manera a su aprendizaje. LaTeX es un ejemplo de software libre.

Además, el uso de software libre podría ser una buena opción para instituciones educativas, como la U.N.A.M., ya que frecuentemente es más económica una licencia de software libre (en caso que se necesite) que una licencia de software comercial.

El trabajo aquí es buscar qué tipo de software libre puede ayudar a la enseñanza en la carrera de Matemáticas Aplicadas y Computación, o software libre que sea adecuado para los proyectos de la Universidad.

Diseño gráfico

La calidad de los documentos generados con LaTeX es tan buena, que muchas editoriales lo utilizan en sus publicaciones. (La)TeX puede ser una herramienta muy útil para quienes estudian diseño gráfico, principalmente los especializados en diseño editorial.

Una herramienta que también podría ser útil para los diseñadores es METAFONT. METAFONT es un lenguaje (creado también por Knuth) que sirve para diseñar tipografía. Gracias a este lenguaje, (La)TeX puede ser utilizado en todo el mundo, debido a que se han creado tipos de letra para varios idiomas, entre ellos, ruso, griego, hebreo, chino y japonés.

Por otra parte, las herramientas PSTricks pueden ser de utilidad para crear gráficos vectoriales en formato PostScript. Otra opción es utilizar METAPOST, un lenguaje basado en METAFONT que también genera gráficos vectoriales. Estas herramientas trabajan sin problemas con (La)TeX.

Apéndice A

Configuración de LaTeX

En este apéndice se explicará brevemente cómo instalar y configurar LaTeX en una computadora con WINDOWS o con LINUX. De igual forma se explicará cómo utilizar algunos programas complementarios para crear archivos PostScript (`.ps`) y PDF (`.pdf`) de un archivo DVI (`.dvi`), cosa que resulta muy útil cuando se está utilizando la clase `prosper` para crear una presentación (sección 2.4).

A.1. Sistema Windows

A.1.1. Instalación

Para instalar LaTeX en una computadora con sistema WINDOWS se necesitarán varios programas, estos son:

1. Compilador de (La)TeX. MiKTeX es el más común.
2. Visualizador de archivos PostScript. Existen AFPL GHOSTSCRIPT y GSVIEW que trabajan juntos para interpretar y visualizar este lenguaje.
3. Visualizador de archivos PDF. El más conocido es ACROBAT READER de ADOBE.
4. Editor de (La)TeX. Existen varios, por ejemplo TeXNICCENTER o WINEDIT.

Todos estos programas son gratuitos, a excepción de WINEDIT, y pueden descargarse de Internet.

La instalación deberá ser como sigue[1]:

1. Instalar MiKTeX. Existen tres tipos de instalación: «Small», «Large» y «Total»; la instalación «Small» es suficiente.
2. Si se trabajará con archivos PostScript es recomendable instalar AFPL GHOSTSCRIPT y GSVIEW; el primero es un interprete del lenguaje y el segundo es un visualizador.
3. De igual manera, si se trabajará con archivos PDF se puede instalar ACROBAT READER.
4. Por último, para facilitar la edición de los documentos se puede instalar TeXNICCENTER.

[1] Para ilustrar el proceso de instalación se utilizarán los programas mencionados anteriormente.

A.1.2. Configuración

En esta sección sólo se tratarán dos aspectos importantes en la configuración de MiKTeX: el idioma y la instalación de paquetes o clases, si existe algún otro problema se puede consultar la documentación del programa.

Idioma

El principal problema cuando se instala MiKTeX es que utilice la división silábica especificada con el paquete babel en cualquier documento; siempre aparecerá un aviso diciendo que la división silábica para el lenguaje *x* no está disponible. Para solucionar este problema se debe hacer lo siguiente:

1. En el menú inicio localizar la carpeta de programas creada por MiKTeX, comúnmente se llama MiKTeX.
2. Ejecutar el programa «MiKTeX Options».
3. Seleccionar la pestaña «Languages».
4. Activar la casilla «spanish» (o el idioma o idiomas que se utilizarán).
5. Aceptar los cambios.
6. Después de esto, MiKTeX manda un aviso diciendo que tendrá que reconstruir los archivos de formato; se acepta.

Una vez hecho esto no habrá problemas con la división silábica para los idiomas que se hayan activado.

Es importante mencionar que no es recomendable desactivar el idioma «english» ni las opciones «dumylang» y «nohyphenation».

Nuevos archivos

Como se mencionó en el capítulo 2, (LA)TeX es un lenguaje de programación, y como tal es posible agregar nuevos paquetes o clases, ampliando así su potencia. En esta sección se explicará cómo instalar este tipo de archivos, como es el caso de las plantillas realizadas en este trabajo.

Cuando se descarga alguna clase[2] es común encontrarse con dos archivos: uno con extensión `.dtx` y otro con extensión `.ins`. Para generar el archivo `.cls` (o `.sty` en el caso de los paquetes) que contiene el código para LaTeX se tendrá que hacer lo siguiente (se tomará como ejemplo la clase aca-report):

1. Ambos archivos deberán estar en el mismo directorio.
2. En el símbolo del sistema se escribe lo siguiente:

   ```
   latex aca-report.ins
   ```

   Esto generará varios archivos, los principales son `aca-report.cls` y `aca-report.drv`.
3. Para generar la documentación se escribe:

[2] El mismo procedimiento aplica para los paquetes.

```
latex aca-report.drv
```

Lo cual generará el archivo `aca-report.dvi`

4. Los archivos que se utilizarán son el de extensión `cls` y `dvi`, los demás pueden eliminarse (excepto los originales).

Una vez que se tiene el archivo `cls` y el `dvi` se deberán copiar a sus carpetas correspondientes:

- En la carpeta `C:\..\texmf\tex\latex` se crea una carpeta (para el ejemplo sería `aca-classes`) y se copia el archivo `cls`.
- En la carpeta `C:\..\texmf\doc\latex` se crea la misma carpeta y se copia el archivo `dvi`.

Por último se tendrá que actualizar la base de datos de MiKTEX. Para hacer esto se abre «MiKTeX Options» y se da clic en el botón que dice «Refresh Now».

A.2. Sistemas Linux

A.2.1. Instalación

La gran ventaja de utilizar algún sistema LINUX es que casi todos los programas necesarios para utilizar LATEX vienen con el sistema: el compilador de (LA)TEXque utilizan es TETEX y existen varios visualizadores de archivos PostScript y PDF dependiendo de la versión del sistema. Lo único que no viene con el sistema es el editor, pero puede utilizarse KILE que funciona parecido a TEXNICCENTER y puede descargarse de Internet.

A.2.2. Configuración

De igual manera que para los sistemas WINDOWS, únicamente se explicará la configuración del idioma y la instalación de paquetes o clases.

Idioma

Para activar la división silábica del idioma español (o cualquier otro) se tendrá que realizar lo siguiente:

1. Buscar el archivo `language.dat`, comúnmente se encuentra en

   ```
   /usr/share/texmf/tex/generic/config/.
   ```

2. Buscar la línea que dice `%!␣spanish` y borrar `%!␣`[3]. Hacer lo mismo para los idiomas que se quieran agregar.
3. Guardar los cambios en el archivo.

Después de hacer las modificaciones es necesario «reiniciar» LATEX de la siguiente manera:

[3] El símbolo ␣ representa un espacio en blanco.

1. Ejecutar `initex /usr/share/texmf/tex/latex/base/latex.ltx` para LaTeX.
2. Ejecutar `pdfinitex /usr/share/texmf/pdftex/latex/config/pdflatex.ini`, si se compilará con pdfLaTeX.

Las rutas pueden variar de una versión a otra; en este caso se utilizó Fedora 2.

Una vez realizado lo anterior, la división silábica del idioma español (o los que se hayan activado) estará disponible.

Nuevos Archivos

El procedimiento para instalar paquetes o clases es el mismo que para un sistema WINDOWS, excepto en el último paso.

Para actualizar la base de datos únicamente se debe ejecutar `mktexlsr`.

A.3. Presentaciones con prosper

Por último, en este apéndice se explicará cómo crear un archivo `pdf` utilizando la clase `prosper`. Este procedimiento, como se mencionó en la sección 2.4.2, es largo, ya que no puede utilizarse directamente `pdflatex`. Dicho procedimiento es el siguiente:

1. Generar el archivo `dvi` con `latex`.
2. Convertir el archivo `dvi` a formato PostScript.
3. Convertir el archivo PostScript a `pdf`.

Los dos últimos puntos se explican a continuación.

Formato PostScript

Las distintas versiones de LaTeX incluyen el programa `dvips` para convertir archivos `dvi` a PostScript. La forma de utilizar este programa es la siguiente:

```
dvips -z archivo.dvi
```

La opción `-z` permite que los «links» funcionen (creados por el paquete `hyperref`).

Si se está trabajando en LINUX a la opción `-z` se tendrá que agregar `-o` para que se genere el archivo PostScript; si sólo se utiliza `-z` el sistema mandará a imprimir el archivo.

Como se mencionó en el capítulo 2, el formato PostScript es el mejor para imprimir, así que si sólo se desea imprimir la presentación hasta aquí es suficiente.

Formato `pdf`

Para proyectar la presentación o distribuirla, lo más conveniente es generar un archivo `pdf`, debido al poco espacio que ocupan.

La conversión de formato PostScript a `pdf` dependerá del software que se disponga para interpretar el lenguaje PostScript. Este software se tendrá que configurar para que convierta los archivos utilizando papel A4, debido a la programación de `prosper`.

Apéndice B

Contenido del CD

Debido al tipo de trabajo, es necesario adjuntar un CD. Este CD contiene las plantillas programadas mencionadas en el capítulo 3, así como los ejemplos de uso del capítulo 4 y el software necesario para utilizar LaTeX en sistemas WINDOWS y LINUX. En este apéndice se detalla el contenido del CD.

La estructura de carpetas y archivos del CD se muestra a continuación, las carpetas se muestran con el símbolo • y los archivos con ○:

- Ejemplos
- Html
- Plantillas
- Programas
  - `inicio.html`
  - `autorun.inf`

La página web `inicio.html` abre la documentación de la carpeta Html, la cual contiene la misma información que se presenta en este apéndice. En sistemas WINDOWS, el archivo `autorun.inf` servirá para abrir automáticamente la página `inicio.html` cuando se carga el CD. El contenido de las carpetas se muestra a continuación.

B.1. Ejemplos

En esta carpeta se encuentran los ejemplos completos utilizados en el capítulo 4. La lista de archivos es la siguiente:

- `DocSoftware.pdf`
- `Manual.pdf`
- `Presentacion.pdf`
- `TInvArticle.pdf`
- `TInvBook.pdf`

En estos se tiene el ejemplo para documentación de software, manual de usuario, presentaciones y trabajos de investigación (con la opción *article* y *book*), respectivamente. El trabajo de titulación no se incluyó, ya que se presenta completo en el presente documento.

Además, en esta carpeta se incluye una carpeta por cada ejemplo. En cada carpeta se incluye todo el código fuente del ejemplo correspondiente. Estos archivos se pueden utilizar como base para crear otros documentos del mismo tipo.

B.2. Html

Esta carpeta contiene la versión electrónica de este apéndice.

B.3. Plantillas

Las plantillas programadas en este trabajo se encuentran en esta carpeta. La lista de archivos es la siguiente:

- `compilado.zip`
- `aca-manual.*`
- `aca-report.*`
- `aca-slides.*`
- `aca-softdoc.*`
- `aca-thesis.*`
- `aca-all.ins`
- `logoUNAM.*`

Cada plantilla está compuesta por dos archivos, uno de extensión `dtx` y otro con extensión `ins` (véase la sección A.1.2 para más detalle), así, por ejemplo, la clase `aca-manual` está compuesta por los archivos `aca-manual.dtx` y `aca-manual.ins`. Si se desea compilar todas las plantillas se puede utilizar el archivo `aca-all.ins`.

Como se mencionó en el capítulo 4, las clases `aca-report` y `aca-thesis` utilizan el logotipo de la U.N.A.M. para componer la portada. En esta carpeta se incluye también dicho logotipo en dos formatos: `eps` y `pdf`, el nombre del archivo es `logoUNAM`. Estos logotipos se deberán colocar en la carpeta `images`, la cual deberá estar en la misma ruta que las plantillas.

Por último, el archivo `compilado.zip` contiene las plantillas compiladas, la documentación de las mismas y los logotipos en su carpeta correspondiente, organizados según la Estructura de Directorios de TEX (TDS). Se recomienda copiar este archivo en la misma ruta donde se encuentra la carpeta `texmf` y descomprimirlo, así todos los archivos necesarios para utilizar las plantillas serán colocados correctamente y únicamente resta actualizar la base de datos del compilador.

B.4. Programas

Esta carpeta contiene los programas necesarios para utilizar LaTeX: compilador, visualizadores y editores. La estructura de esta carpeta es la siguiente:

- Linux
- Windows

El contenido de cada uno se explica a continuación.

B.4.1. Linux

Para utilizar LaTeX en un sistema LINUX no se requiere de algún programa especial, ya que la mayoría de las versiones incluyen un compilador de LaTeX, editores para varios lenguajes de programación y visualizadores para los archivos que se generan con LaTeX. Esta carpeta únicamente incluye lo siguiente:

- `acroread-5.06-1.2mlx.i386.rpm` (`http://www.adobe.com`)
- `kile-1.8b1.tar.bz2` (`http://kile.sourceforge.net`)

El primer archivo es la versión 5.06 de ACROBAT READER de ADOBE, un visualizador de archivos `pdf`. El segundo archivo es la versión 1.8b.1 de KILE, un IDE para (LA)TeX. Al final del nombre del archivo se incluye la dirección en internet del sitio donde puede descargase la última versión del programa.

B.4.2. Windows

Los sistemas WINDOWS necesitan de varios programas para utilizar LaTeX. Esta carpeta contiene lo siguiente:

- Miktex (`http://www.miktex.org`)
- `AdbeRdr60_esp_full.exe` (`http://www.adobe.com`)
- `gs814w32.exe` (`http://www.cs.wisc.edu`)
- `gsv46w32.exe` (`http://www.cs.wisc.edu`)
- `TXCSetup_1Beta6_30.exe` (`http://www.toolscenter.org`)

En la carpeta Miktex se encuentra el archivo `setup-2.4.1661.exe` para instalar la versión 2.4 del compilador MIKTEX. El archivo `AdbeRdr60_esp_full.exe` es la versión 6.0.1 de ACROBAT READER de ADOBE. Los archivos `gs814w32.exe` y `gsv46w32.exe` son un interprete y visualizador de archivos PostScript, respectivamente; el primero es la versión 8.14 de GHOSTSCRIPT y el segundo es la versión 4.6 de GSVIEW. Por último, el archivo `TXCSetup_1Beta6_30.exe` es la versión 1 Beta 6.30 del programa TEXNICCENTER, un IDE para (LA)TeX.

Bibliografía

Anderson, Paul V. 1987. *Technical Writing: A Reader-Centered Approach*. San Diego: Harcourt Brace Jovanovich.

Braude, Eric J. 2003. *Ingeniería de software, una perspectiva orientada a objetos*. Alfaomega.

Cascales Salinas, Bernardo, Pascual Lucas Saorín, José Manuel Mira Ros, Antonio José Pallarés Ruiz y Salvador Sánchez-Pedreño Guillén. 2003. *El libro de LaTeX*. Pearson Educación.

González Videgaray, MariCarmen. 2004. *Mapeo de Información Técnica*.

Goossens, Michel, Frank Mittelbach y Alexander Samarin. 1994. *The LaTeX Companion*. Addison-Wesley.

Knuth, Donald Ervin. 1986. *The TeXbook*. Addison-Wesley.

Lamport, Leslie. 1994. *LaTeX — A document Preparation System. User's guide and manual references*. Segunda edición. Addison-Wesley.

McMurrey, David A. *Online Technical Writing*. Brooklyn College, Brooklyn, New York.

Pfleeger, Shari Lawrence. 2002. *Ingeniería de software, teoría y práctica*. Pearson Educación.

Turabian, Kate L. 1987. *A Manual for Writers of Term Papers, Theses, and Dissertactions*. Quinta edición. The University of Chicago Press.

Wolf, Kurt Bernardo, Gilberto Becerril, Ricardo Espriella, Eumelia Mendoza, Enrique Molina, Miguel Navarro Saad y Martha Pavón. 1986. *Manual de lenguaje y tipografía científica en castellano*. Trillas.

Índice de figuras

Índice de cuadros