puede pulsar en la opción de menú *Source->Format* para formatear el código, véase Figura 1.15 (las reglas de formato incorporadas por defecto en Eclipse cumplen en su mayoría las reglas definidas por Oracle).

Una vez generado el código fuente es necesario traducirlo a un lenguaje que sea entendido por la máquina, para ello se utilizan otros programas llamados **compiladores** o **intérpretes**. El resultado de la traducción es el **código objeto**. Este último aún no es ejecutable por la máquina. Se necesita un programa enlazador o montador que tome el código objeto y lo conecte con una serie de librerías para construir el **código ejecutable**. Una vez obtenido el código ejecutable es necesario probar el programa con el fin de comprobar y verificar que cumple con las especificaciones del diseño.

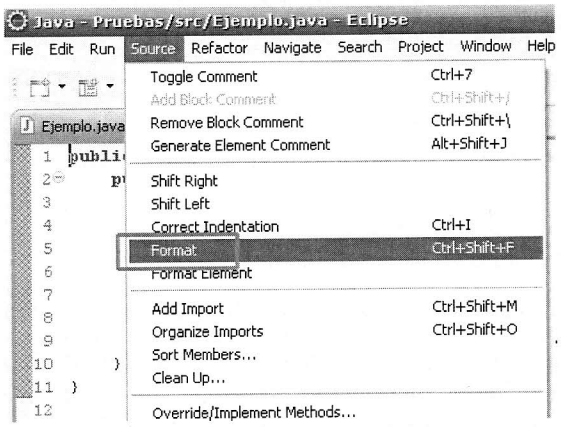

Figura 1.15. Formateo de texto desde Eclipse.

A la vez que se va desarrollando código se deben ir escribiendo los manuales técnicos y de referencia necesarios, así como la parte inicial correspondiente del manual de usuario. Esta documentación es esencial para la etapa de pruebas y mantenimiento, así como para la entrega final del producto.

1.4.4. Pruebas

En esta etapa ya se dispone del software y se trata de encontrar errores, no solo de codificación sino también relativos a la especificación o el diseño. Durante la prueba del software se realizarán tareas de verificación y validación del software (V&V):

- **La verificación**: se refiere al conjunto de actividades que tratan de comprobar si se está construyendo el producto correctamente, es decir, si el software implementa correctamente una función específica.
- **La validación**: se refiere al conjunto de actividades que tratan de comprobar si el producto es correcto, es decir, si el software construido se ajusta a los requisitos del cliente.

El objetivo en esta etapa es planificar y diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo. Una prueba tiene éxito si descubre un error no detectado hasta entonces. Un **caso de prueba** es un documento que especifica los valores de entrada, salida esperada y las condiciones previas para la ejecución de la prueba.

Las recomendaciones para las pruebas son las siguientes:

- Cada prueba debe definir los resultados de salida esperados.
- Un programador o una organización debe evitar probar sus propios programas.
- Es necesario revisar los resultados de cada prueba en profundidad.
- Las pruebas deben incluir datos de entrada válidos y esperados, así como no válidos e inesperados.
- Centrar las pruebas en dos objetivos: 1) comprobar si el software no hace lo que debe hacer y 2) comprobar si el software hace lo que no debe hacer.
- Evitar hacer pruebas que no estén documentadas ni diseñadas con cuidado.
- No planear pruebas asumiendo que no se encontrarán errores.
- La probabilidad de encontrar errores en una parte del software es proporcional al número de errores ya encontrados.
- Las pruebas son una tarea creativa.

El flujo de proceso para probar el software se muestra en la Figura 1.16, es el siguiente:

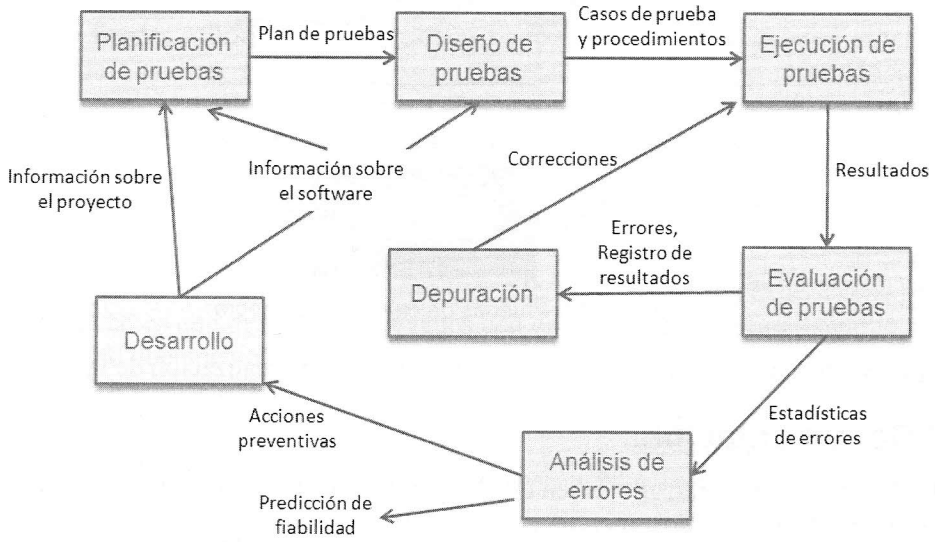

Figura 1.16. Flujo de proceso para probar el software.

1. En primer lugar, hay que **generar un plan de pruebas** partiendo de la documentación del proyecto y de la documentación sobre el software a probar.
2. A partir del plan **se diseñan las pruebas**. Se identifican las técnicas a utilizar para probar el software.
3. **Generación de los casos de prueba**. Se han de confeccionar los distintos casos de prueba según la técnica o técnicas identificadas previamente.
4. **Definición de los procedimientos de la prueba**. Hay que especificar cómo se va a llevar a cabo el proceso, quién lo va a realizar, cuándo, etc.
5. **Ejecución de las pruebas** aplicando los casos de prueba generados previamente.
6. **Evaluación**. Se identifican los posibles errores producidos al comparar los resultados obtenidos en la ejecución con los esperados. Es necesario realizar un informe con el

resultado de ejecución de las pruebas, qué casos de prueba pasaron satisfactoriamente, cuáles no, y qué fallos se detectaron.

7. **Depuración**. Trata de localizar y corregir los errores. Si se corrige un error se debe volver a probar el software para ver que se ha resuelto el problema. Si no se consigue localizar un error puede ser necesario realizar más pruebas para obtener más información.
8. El **análisis de errores** puede servir para predecir la fiabilidad del software y mejorar los procesos de desarrollo.

Para llevar a cabo el diseño de casos de prueba se utilizan dos técnicas: **prueba de caja blanca** y **prueba de caja negra** (véase Figura 1.17). Las primeras se centran en validar la estructura interna del programa (necesitan conocer los detalles procedimentales del código) y las segundas se centran en validar los requisitos funcionales sin fijarse en el funcionamiento interno del programa (necesitan saber la funcionalidad que el código ha de proporcionar). Estas pruebas no son excluyentes y se pueden combinar para descubrir diferentes tipos de errores.

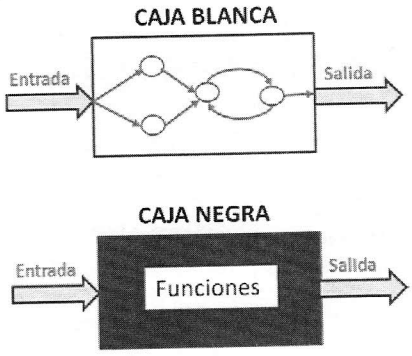

Figura 1.17. Pruebas de caja blanca y negra.

En el Capítulo 3 se tratarán más ampliamente el diseño y la realización de las pruebas.

1.4.5. Documentación

Todas las etapas del desarrollo deben quedar perfectamente documentadas. En esta etapa será necesario reunir todos los documentos generados y clasificarlos según el nivel técnico de sus descripciones.

Los documentos relacionados con un proyecto de software:

- Deben actuar como un medio de comunicación entre los miembros del equipo de desarrollo.
- Deben ser un repositorio de información del sistema para ser utilizado por el personal de mantenimiento.
- Deben proporcionar información para ayudar a planificar la gestión del presupuesto y programar el proceso del desarrollo del software.
- Algunos de los documentos deben indicar a los usuarios cómo utilizar y administrar el sistema.

Por lo general se puede decir que la documentación presentada se divide en dos clases:

- **La documentación del proceso**. Estos documentos registran el proceso de desarrollo y mantenimiento. Son documentos en los que se indican planes, estimaciones y horarios que se utilizan para predecir y controlar el proceso de software, que informan sobre cómo usar los recursos durante el proceso de desarrollo, sobre normas de cómo se ha de implementar el proceso.
- **La documentación del producto**. Esta documentación describe el producto que está siendo desarrollado. Define dos tipos de documentación: la documentación del sistema que describe el producto desde un punto de vista técnico, orientado al desarrollo y mantenimiento del mismo; y la documentación del usuario que ofrece una descripción del producto orientada a los usuarios que utilizarán el sistema.

La documentación del proceso se utiliza para gestionar todo el proceso de desarrollo del software. La documentación del producto se utiliza después de que el sistema está en funcionamiento, aunque también es esencial para la gestión del desarrollo del sistema.

DOCUMENTACIÓN DEL USUARIO

Los usuarios de un sistema no son todos iguales. Hay que distinguir entre los usuarios finales y los usuarios administradores del sistema.

Los usuarios finales utilizan el software para realizar alguna tarea, quieren saber cómo el software les ayuda a realizar su tarea, no están interesados en el equipo informático o en los detalles del programa. **Los administradores del sistema** son responsables de la gestión de los programas informáticos utilizados por los usuarios finales, pueden actuar como un gestor de red si el sistema implica una red de estaciones de trabajo o como un técnico que arregla problemas de software de los usuarios finales y que sirve de enlace entre los usuarios y el proveedor de software.

Para atender a los distintos tipos de usuarios con diferentes niveles de experiencia, se definen una serie de documentos que deben ser entregados con el sistema de software (aunque se pueden entregar en el mismo documento), véase Figura 1.18:

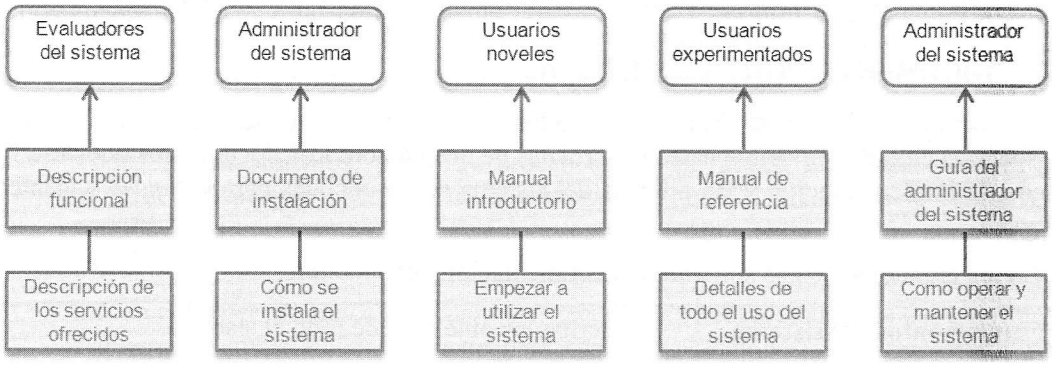

Figura 1.18. Diferentes tipos de documentación de usuario.

Descripción funcional del sistema	Proporciona una descripción general de las funciones del sistema. Los usuarios deben ser capaces de leer este documento como un manual introductorio y decidir si el sistema es lo que necesitan

Documento de instalación del sistema	Está destinado a los administradores del sistema. Describe cómo instalar el sistema en un entorno particular. Debe contener una descripción de los ficheros que componen el sistema, su configuración, el hardware mínimo requerido, cómo iniciar el sistema, etc.
Manual introductorio.	Proporciona explicaciones sencillas de cómo empezar a utilizar el sistema. Para facilitar las explicaciones se deben usar ejemplos. Los usuarios principiantes cometen errores, se debe describir la manera de recuperarse ante ellos
Manual de referencia del sistema	Proporciona la descripción detallada de la instalación del sistema y su uso. Debe proporcionar una lista completa de mensajes de error y debe describir cómo recuperarse ante los errores detectados. Debe ser completo, formal y descriptivo
Guía del administrador del sistema	Para algunos sistemas donde hay que introducir comandos de control se deben indicar las tareas del operador, se deben describir los mensajes que se producen y las respuestas que el operador tiene que dar al sistema. Por ejemplo, se deben describir los mensajes generados cuando el sistema interactúa con otros sistemas
Tarjeta de referencia rápida	Sirven para hacer búsquedas rápidas. Por ejemplo, un sistema de ayuda en línea que contiene información breve sobre alguna parte del sistema, permite al usuario ahorrar el tiempo invertido en consultar manuales (aunque no debe ser visto como sustituto para la documentación más completa)

DOCUMENTACIÓN DEL SISTEMA

La documentación del sistema incluye todos los documentos que describen el sistema, desde la especificación de requisitos hasta las pruebas de aceptación. Los documentos que describen el diseño, la implementación y las pruebas del sistema son esenciales para entender y mantener el software.

En los grandes sistemas software la documentación debe incluir:

Fundamentos del sistema	Describe los objetivos de todo el sistema
El análisis y especificación de requisitos.	Proporciona información exacta sobre la especificación de los requisitos, según lo acordado entre las partes interesadas (usuarios, clientes, desarrollador)
Diseño	Describe la arquitectura del sistema: cómo se aplican los requisitos del sistema, la forma en que se descompone el sistema en un conjunto de unidades de programas que interactúan y la función de cada unidad de programa

Implementación	Proporciona la descripción de: la forma en que se expresa el sistema detallado en algún lenguaje de programación formal y las acciones del programa en forma de comentarios dentro de los programas
Plan de pruebas del sistema	Proporciona una descripción de cómo las unidades de programa son evaluadas individualmente y como todo el sistema se prueba después de la integración
Plan de pruebas de aceptación	Se describen las pruebas que el sistema debe pasar antes de que los usuarios las acepten
Los diccionarios de datos	Contiene las descripciones de todos los términos que se relacionan con el sistema de software en cuestión

Para los sistemas más pequeños, la documentación suele ser menos amplia, como mínimo se debe tratar de mantener la especificación del sistema, el documento de diseño arquitectónico y el código fuente del programa.

Por desgracia, el mantenimiento de la documentación a menudo se descuida y puede no estar en sintonía con el software asociado, causando problemas para los usuarios y mantenedores del sistema. La mejor solución a este problema consiste en apoyar el mantenimiento de documentos con herramientas software que registran los documentos, las relaciones entre ellos, avisan a los desarrolladores cuando hay cambios en un documento, registran posibles inconsistencias en la documentación.

ESTRUCTURA DEL DOCUMENTO

La documentación debe estar correctamente organizada para permitir a los lectores localizar la información más fácilmente, organizada en capítulos, y dentro de estos capítulos, en secciones y subsecciones. La estructura del documento tiene un impacto importante en la legibilidad y usabilidad. Algunas pautas mínimas de estructuración son las siguientes:

- Todos los documentos, por muy breves que sean, deben tener una portada que identifica el proyecto, el tipo de documento, el autor, la fecha de la creación, la versión, los revisores, los destinatarios del documento y la clase de la confidencialidad del documento.
- Deben contener un índice con capítulos, secciones y subsecciones. Se debe definir un esquema de numeración coherente para capítulos, secciones y subsecciones y para numerar las páginas (el número de página debe ser capítulo – página).
- Se debe incluir al final un glosario de términos ya que el documento está dirigido a un amplio espectro de lectores que pueden tener diferentes vocabularios.

El estándar IEEE para la documentación del usuario (IEEE , 2001) propone que la estructura de un documento debe incluir los componentes que se muestran en la tabla:

COMPONENTE	DESCRIPCIÓN
Datos de identificación	Datos como el título y el identificador que únicamente identifican el documento
Tabla de contenidos	Capítulo, nombres de las secciones y los números de página
Lista de ilustraciones	Números de figura y títulos
Introducción	Define el propósito del documento y un breve resumen de los contenidos
Información para el uso de la documentación	Sugerencias para diferentes lectores sobre cómo utilizar la documentación con eficacia
Conceptos de las operaciones	Una explicación de la base conceptual para el uso del software
Procedimientos	Las instrucciones sobre cómo utilizar el software
Información sobre los comandos software	Una descripción de cada uno de los comandos soportados por el software
Mensajes de error y resolución de problemas	Una descripción de los mensajes de error que pueden aparecer y cómo recuperarse de ellos
Glosario	Las definiciones de términos especiales utilizados
Fuentes de información relacionadas	Las referencias o enlaces a otros documentos que proporcionan información adicional
Características de navegación	Las características que permiten a los lectores encontrar su actual ubicación y moverse por el documento
Índice	Una lista de términos clave y las páginas en las que estos términos se referencian
Capacidad de búsqueda	En la documentación electrónica, indica la manera de encontrar términos específicos en la documentación

1.4.6. Explotación

Una vez que se han realizado todas las pruebas y documentado todas las etapas, se pasa a la explotación del sistema. En esta etapa se lleva a cabo la instalación y puesta en marcha del producto software en el entorno de trabajo del cliente.

En esta etapa se llevan a cabo las siguientes tareas:

- **Se define la estrategia para la implementación del proceso.** Se desarrolla un plan donde se establecen las normas para la realización de las actividades y tareas de este proceso. Se definen los procedimientos para recibir, registrar, solucionar, hacer un seguimiento de los problemas y para probar el producto software en el entorno de trabajo.

- **Pruebas de operación.** Para cada release del producto software, se llevarán a cabo pruebas de funcionamiento y tras satisfacerse los criterios especificados, se libera el software para uso operativo.
- **Uso operacional del sistema.** El sistema entrará en acción en el entorno previsto de acuerdo con la documentación de usuario.
- **Soporte al usuario.** Se deberá proporcionar asistencia y consultoría a los usuarios cuando la soliciten. Estas peticiones y las acciones subsecuentes se deberán registrar y supervisar.

1.4.7. Mantenimiento

El mantenimiento del software se define (*Norma IEEE 1219 para Mantenimiento de Software*) como la modificación de un producto de software después de la entrega para corregir los fallos, para mejorar el rendimiento u otros atributos, o para adaptar el producto a un entorno modificado.

Existen cuatro tipos de mantenimiento del software que dependen de las demandas de los usuarios del producto software a mantener (véase Figura 1.19):

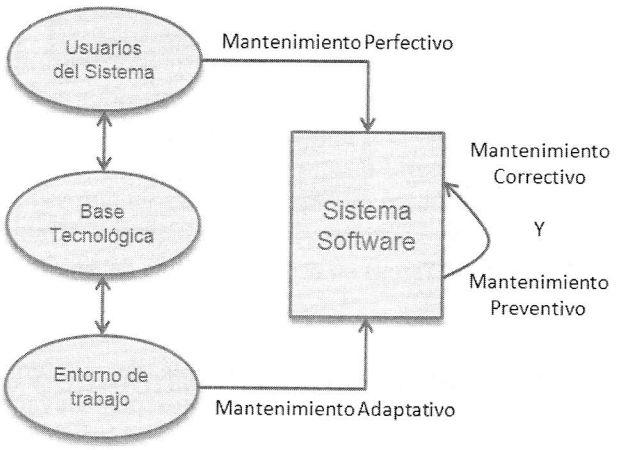

Figura 1.19. Tipos de mantenimiento del software.

- **Mantenimiento adaptativo.** Con el paso del tiempo es posible que se produzcan cambios en el entorno original (CPU, sistema operativo, reglas de la empresa, etc.) para el que se desarrolló el software. El mantenimiento adaptativo tiene como objetivo la modificación del producto por los cambios que se produzcan, tanto en el hardware como en el software del entorno en el que se ejecuta. Este tipo de mantenimiento es el más usual debido a los rápidos cambios que se producen en la tecnología informática, que en la mayoría de ocasiones dejan obsoletos los productos software desarrollados.

- **Mantenimiento correctivo.** Es muy probable que después de la entrega del producto, el cliente encuentre errores o defectos, a pesar de las pruebas y verificaciones realizadas en las etapas anteriores. Este tipo de mantenimiento tiene como objetivo corregir los fallos descubiertos.

- **Mantenimiento perfectivo.** Conforme el cliente utiliza el software puede descubrir funciones adicionales que le pueden aportar beneficios. El mantenimiento perfectivo es la modificación del producto de software orientado a incorporar nuevas funcionalidades (más allá de los requisitos funcionales originales) y nuevas mejoras en el rendimiento o la mantenibilidad del producto.

- **Mantenimiento preventivo.** Consiste en la modificación del producto de software sin alterar las especificaciones del mismo, con el fin de mejorar y facilitar las tareas de mantenimiento. Este tipo de mantenimiento hace cambios en programas con el fin de que se puedan corregir, adaptar y mejorar más fácilmente, por ejemplo, se pueden reestructurar los programas para mejorar su legibilidad o añadir nuevos comentarios que faciliten la comprensión del mismo. A este mantenimiento también se le llama *reingeniería del software.*

Las tareas que se llevan a cabo en esta etapa son (véase Figura 1.20):

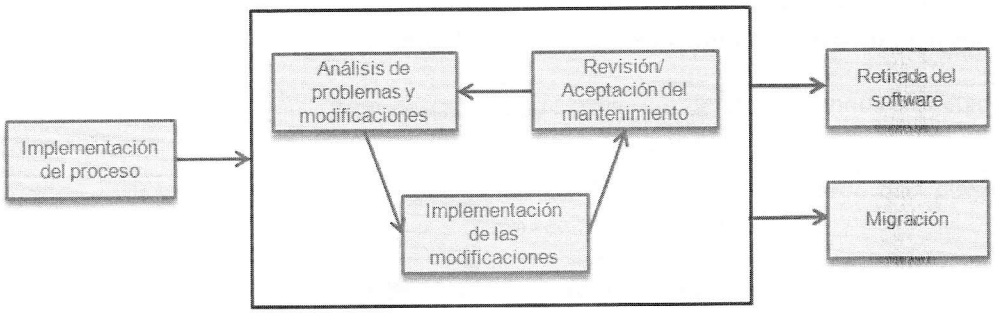

Figura 1.20. Tareas en el mantenimiento del software.

- **Implementación del proceso.** El responsable de mantenimiento deberá preparar, documentar y ejecutar planes y procedimientos para llevar a cabo las actividades y tareas del proceso de mantenimiento. Deberá establecer procedimientos para recibir, registrar y hacer seguimiento a los informes de problemas y a las peticiones de modificaciones de los usuarios y proporcionar información a los usuarios sobre su situación.

- **Análisis de problemas y modificaciones.** El responsable de mantenimiento deberá analizar el informe del problema o la petición de modificación, de acuerdo con su impacto en la organización, el sistema existente y los sistemas con los que interacciona. Deberá verificar el problema, desarrollar opciones para la implementación de la modificación y documentar la solicitud de modificación, los resultados del análisis y las opciones de implementación.

- **Implementación de las modificaciones.** El responsable de mantenimiento tiene que llevar a cabo el análisis y determinar qué documentación, unidades de software y versiones se deben ser modificar, es preciso documentar este análisis. Deberá ejecutar el proceso de desarrollo para implementar las modificaciones.

- **Revisión/aceptación del mantenimiento**. El responsable de mantenimiento deberá llevar a cabo revisiones, con la organización que autoriza las modificaciones, para determinar la integridad del sistema modificado. Deberá obtener aprobación para la finalización satisfactoria de la modificación.

- **Migración.** Se deberá preparar, documentar y ejecutar un plan de migración. Las actividades de planificación deberán incluir a los usuarios a los que se les dará notificación de los planes y actividades de migración.

- **Retirada del software.** El producto software se retirará por petición del propietario. Se deberá preparar y documentar un plan de retirada para el cese del soporte activo por parte de las organizaciones de operación y mantenimiento. Se deberá notificar a los usuarios los planes y actividades de la retirada. Para facilitar la transición al nuevo sistema, conviene que se lleve a cabo la operación en paralelo del sistema a retirar y del nuevo producto software. Durante este período, se deberá proporcionar formación a los usuarios.

1.5. CONCEPTO DE PROGRAMA

Un **programa informático** es un conjunto de instrucciones escritas en un lenguaje de programación que aplicadas sobre un conjunto de datos resuelven un problema o parte del mismo. Para que el programa pueda ser ejecutado es necesario traducirlo a un lenguaje entendible por el ordenador, el lenguaje máquina; esta tarea se lleva a cabo por otro programa llamado compilador. Una vez que tenemos el programa en código entendible por la máquina hay que cargarlo en la memoria principal para que el procesador ejecute una a una todas las instrucciones.

1.5.1. Programa y componentes del sistema informático

Para ejecutar un programa se necesitan los recursos hardware del ordenador: el procesador (también conocido como UCP *Unidad Central de Proceso* o CPU), la memoria RAM (o memoria principal), los dispositivos de E/S, etc. Las instrucciones de un programa se cargan en la memoria principal y la CPU será la encargada de ejecutarlas.

En una arquitectura Von Neumann, véase Figura 1.21, son varios los componentes que forman la CPU:

- **La Unidad de Control (UC)** que interpreta y ejecuta las instrucciones máquina almacenadas en la memoria principal y genera las señales de control necesarias para ejecutarlas.

- **La Unidad Aritmético-Lógica (UAL)**, recibe los datos sobre los que efectúa operaciones de cálculo y comparaciones, y toma decisiones lógicas (determina si una afirmación es cierta o falsa mediante las reglas del álgebra de Boole), y devuelve el resultado, todo ello bajo la supervisión de la Unidad de Control.

- **Los registros** de trabajo o de propósito general donde se almacena información temporal, es el almacenamiento interno de la CPU.