Los **DIAGRAMAS DE ESTADO** explican todos los estados posibles por

Por Ej:

-Verdadero o Falso

-Mensaje

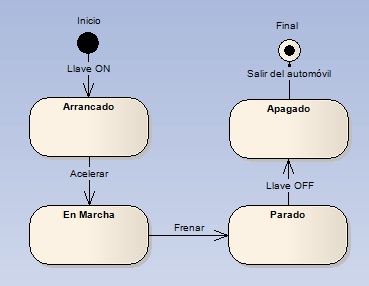
-Paso de cierto período de tiempo después de entrar al estado o de cierta hora y fecha particular

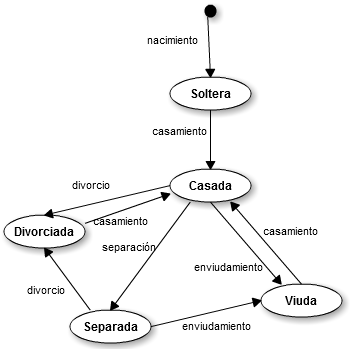
-El nombre de un evento tiene alcance dentro del paquete en el cual está definido, no es local a la clase que lo nombre.

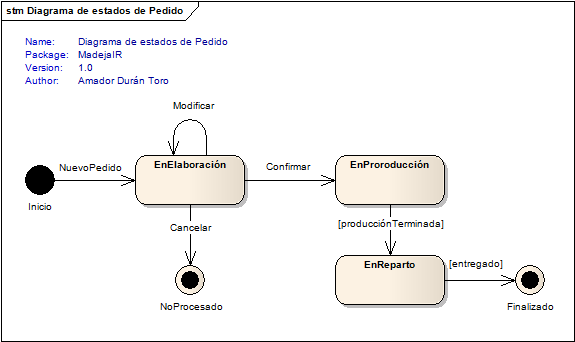
Una transición es una relación entre dos estados que indica que un objeto que esté en el primer estado entrará en el segundo estado cuando ocurra un evento especificado.

El DIAGRAMA DE ESTADO:

* Determina todas las rutas o caminos que puede tomar la información luego de ejecutarse cada proceso.
* Son útiles para modelar la vida de un objeto
* Permite visualizar de una forma ordenada la ejecución de cada uno de los procesos.
* Se indica qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera.
* Permite que el analista se centre en las necesidades del usuario, ya que expresa la intención que tiene el actor (su usuario) al hacer uso del sistema.









CONCLUSIÓN

Un Diagrama de Estados sirve para mostrar la vida de un objeto.

El diagrama indica los eventos que causan que un estado cambie a otro y cuáles son las respuestas y acciones que genera este.

El diagrama de estado se utiliza normalmente para describir objetos del dominio del usuario y se documenta por lo general en la etapa de análisis.

**HERRAMIENTAS UML**

Es difícil decir que una herramienta es la mejor. La decisión depende de muchos factores, no solo el que cumplan al pie de la letra la especificación de UML, sino además, que sea una herramienta intuitiva y que no consuma demasiados recursos de nuestro ordenador.

Para decidir que herramienta UML utilizar analizamos algunas características como:

-Código abierto

-Tipo de licencia (que hubiera un licenciamiento académico a un coste accesible)

-Integración con entornos de desarrollo (y cuales)

-Coste

-Versión de UML

-Diagramas que soporta

-Generación de código (y que lenguajes de programación soporta)

-Capacidades de ingeniería inversa

-Sistema Operativo

-Requisitos de instalación

Las herramientas analizadas fueron:

-ArgoUML

-Poseidon for UML

-OpenAmeos

-StarUML

-Enterprise Architect

-Umbrello UML Modeler

-UML Designer

-*Visual Paradigm for UML*

-*Rational Software Modeler (sucesor de Rational Rose)*