**UNIVERSIDAD DE ORIENTE**

**NUCLEÓ ANZOÁTEGUI**

**ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS**

**DEPARTAMENTO DE COMPUTACION Y SISTEMAS**

**DESARROLLO DE SOFTWARE**



**UML 2.0 PARTE II**

**Estudiantes:**

Correa, Luis. C.I: 19.840.230 Ingeniería En Computación

Dun, Manuel. C.I: 19.257.821 Ingeniería En Computación

Medina, Orlando. C.I: 16.067.176 Ingeniería En Computación

**PROFESOR:**

Ing. Víctor Mujica.

**BARCELONA 28 DE ENERO DE 2015**

**INDICE**

**Pág.**

**1.** **INTRODUCCIÓN:** 3

**2.** **DEFINICIÓN DE UML:** 4

**3.** **DIAGRAMA DE COMUNICACIÓN** 7

**4.** **DIAGRAMA DE ESTADO** 10

**5.** **DIAGRAMA DE PAQUETES** 12

**6.** **DIAGRAMA DE DESPLIEGUE** 14

**7.** **DIAGRAMA DE ACTIVIDAD** 15

**8.** **DIAGRAMA DE TIEMPO:** 20

**9.** **DIAGRAMA DE VISIÓN GLOBAL DE INTERACCIONES:** 21

**10.** **CONCLUSIONES**: 23

.

1. **INTRODUCCIÓN:**

Los diagramas UML 2.0 son utilizados ampliamente en el campo de la ingeniería del software ya que brindan un apoyo al desarrollador para poder entender mejor lo que va a crear, el UML 2.0 también les brinda apoyo a los futuros trabajadores del software para entender mejor el programa.

En la era de la información los ingenieros en computación han tenido que adaptarse a las nuevas tecnologías para el desarrollo del software el UML 2.0 es un ejemplo de ello.

Los ingenieros han tenido que adaptarse a las nuevas técnicas utilizadas para el desarrollo de software.

La tendencia obligada para un profesional de computación es tener el conocimiento y manejo de varios tipos de diagramas y que mejor que utilizar el UML 2.0, en el cual se utilizan múltiples diagramas.

1. **DEFINICIÓN DE UML:**

UML es un lenguaje de modelado visual que sirve para:

* **Visualizar**
* **Especificar**
* **Construir**
* **Documentar**

Trabaja modelando sistemas, independientemente de la metodología de análisis y diseño, siempre y cuando se realice bajo una perspectiva orientada a objeto

Es un modelo explícito, que facilita la comunicación y a la construcción de modelos precisos, no ambiguos y completos.



**CONCEPTOS DE MODELO**

* SISTEMA: colección de elementos, posiblemente divididos en subsistemas, organizados para lograr un propósito. Está descrito por un conjunto de modelos.
* MODELO: simplificación completa y auto consistente de la realidad, creado para comprender un sistema.
* VISTA (Arquitectural): es una proyección de la organización y estructura de un modelo de un sistema, centrada en su aspecto. Incluye un subconjunto de los elementos en el modelo.
* DIAGRAMA: representación gráfica de un conjunto de elementos del modelo y sus relaciones. En UML generalmente corresponde a un grafo conexo de nodos (Objetos) y sus arcos (Relaciones).

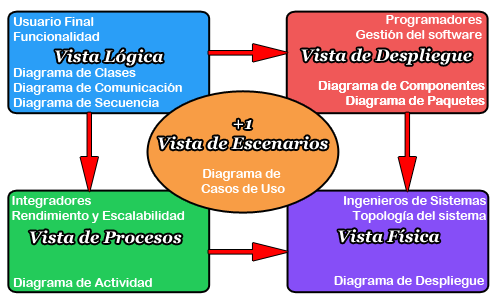
Un modelo captura las propiedades estructurales y de comportamiento de un sistema, es una abstracción de dicho sistema. El modelo describe completamente aquellos aspectos del sistema que son relevantes al propósito del modelo

**VISTAS ARQUITECTURALES**

Durante el desarrollo de un sistema de Software se requiere que este sea visto desde varias perspectivas

Diferentes usuarios miran el sistema de formas diferentes en momentos diferentes. La arquitectura del sistema es clave para poder manejar estos puntos de vista diferentes:

* Se organiza mejor a través de vistas arquitecturales interrelacionadas
* Proyecciones del modelo del sistema centrada en un aspecto particular.



**VISTA DE ESCENARIOS**

Captura la funcionalidad del sistema tal y como es percibido por los usuarios finales. Los diagramas correspondientes a esta vista son el diagrama de Casos de Uso

**VISTA LÓGICA**

Captura las clases, las interfaces, y colaboraciones que describen al sistema:

* En el dominio del problema
* En el dominio de la solución

Los elementos que la conforman dan soporte a los requisitos funcionales del sistema. Los diagramas correspondientes son los diagramas de clase, secuencia y comunicación.

**VISTA DE PROCESOS**

Captura el flujo de control entre las diversas partes del sistema, incluyendo los posibles mecanismos de concurrencia y sincronización. Abarca en especial requisitos no funcionales el rendimiento, la escalabilidad y capacidad de procesamiento

Los diagramas correspondientes son los diagramas de actividad

**VISTA FÍSICA**

Captura las características de instalación y ejecución del sistema. Contiene los nodos y enlaces que forman la topología hardware sobre la que se ejecuta el sistema software

Se ocupa principalmente de la distribución de las partes que forman el sistema software real. Los diagramas que corresponden es el diagrama de despliegue.

**VISTA DE DESPLIEGUE**

Captura los artefactos que se utilizan para ensamblar y poner en producción el sistema software real. Aquí se define la arquitectura física del sistema. Corresponde en esta vista los diagramas de paquetes y de componentes

Cada vista puedes existir de forma independiente, pero actúan entre sí los nodos

En resumen, el modelo UML es un conjunto de elementos de modelado que definen la estructura, el comportamiento y la funcionalidad del sistema. La presentación de estos conceptos se realiza a través de múltiples diagramas con el fin de hacerlo comprensibles. Estos diagramas se agrupan en vistas, cada una enfocada a un aspecto particular del sistema.

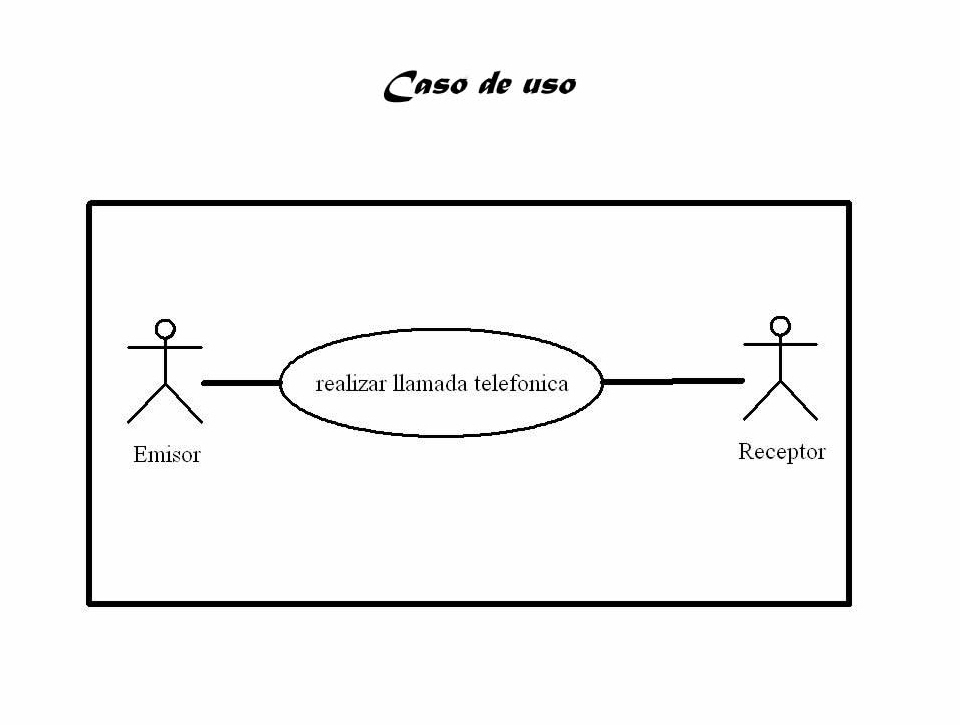
Un diagrama es una representación gráfica de un conjunto de elementos que con sus relaciones, ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas.

1. **DIAGRAMA DE COMUNICACIÓN**

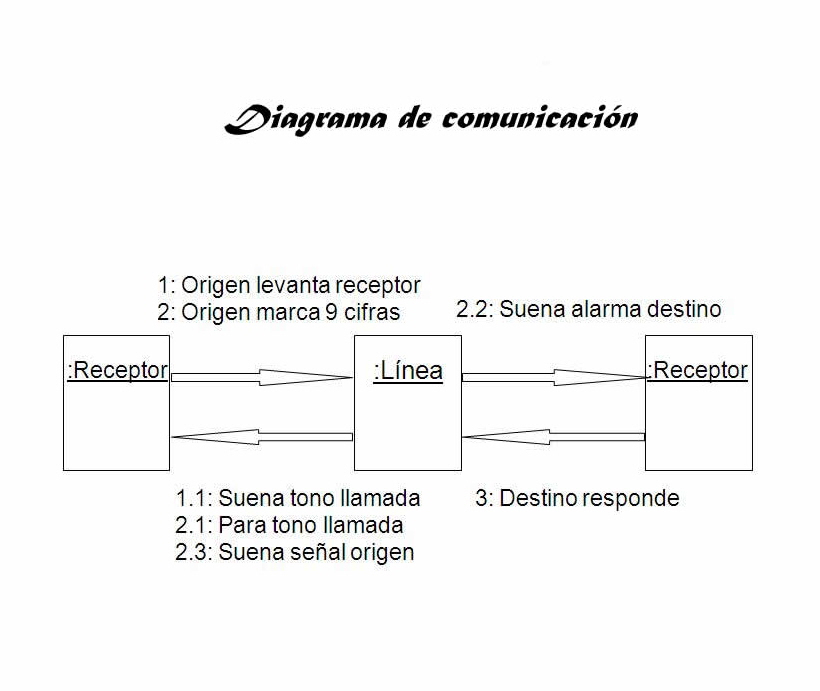
Un diagrama de comunicación es una versión simplificada del diagrama de colaboración de la versión de UML 1.X

Este diagrama describe las interacciones entre los objetos en términos de mensajes secuenciales. Es una forma interactiva del diagrama de secuencias. Existen herramientas que automáticamente convierten estos diagramas el uno en el otro

Este diagrama representa una combinación de información tomada desde el diagrama de clases, secuencial y el diagrama de caso de uso, describiendo tanto la estructura estática como el comportamiento dinámico de un sistema

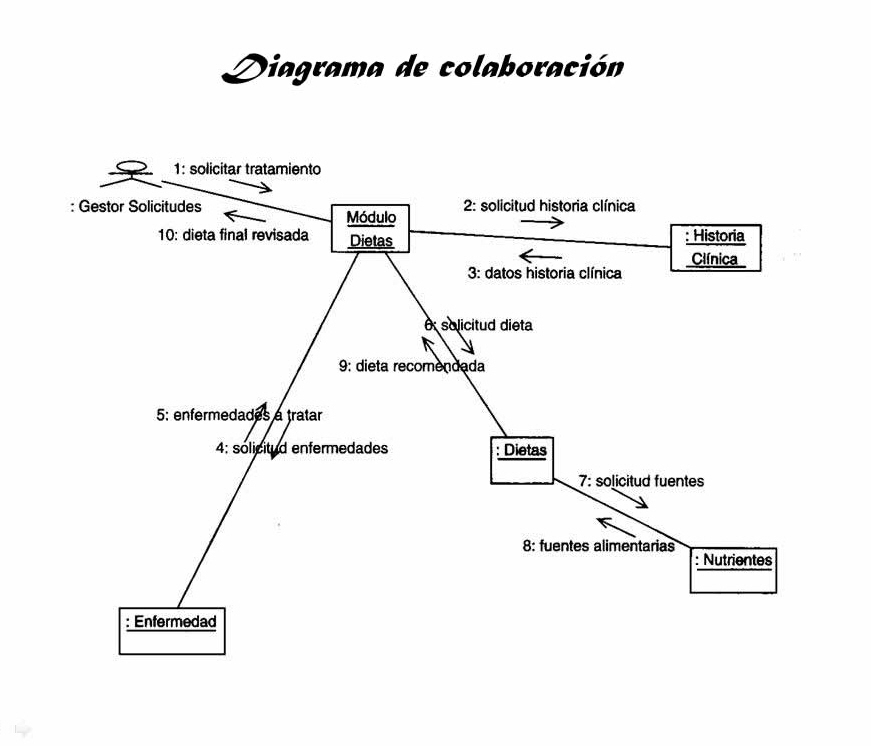


Para mantener el orden de los mensajes en un diagrama de comunicación, estos son etiquetados con un número cronológico y colocado cerca del enlace por el cual se desplaza el mensaje.



Leer un diagrama de comunicación conlleva comenzar en el mensaje 1.0 y seguir los mensajes desde un objeto hasta el siguiente sucesivamente. Algunas de sus ventajas son que si el esquema es simple, el comportamiento también será simple. Existe una equivalencia semántica, además de que este diagrama muestra claramente los objetos que están conectados a un determinado objeto. Su debilidad es que no muestra el orden en el que se ejecutan los mensajes tan bien como el diagrama de secuencia. Elementos:

* Instancias de clases: Rectángulo con el nombre de la instancia subrayado.
* Mensajes: Los mensajes se muestran como flechas etiquetadas unidas a los enlaces. Cada mensaje tiene un número de secuencia, una lista opcional de mensajes precedentes, una condición opcional de guarda, un nombre, una lista de argumentos y un nombre de valor de retorno opcional.
* Flujos: Generalmente, un diagrama de comunicación contiene un símbolo para un objeto durante una operación completa.

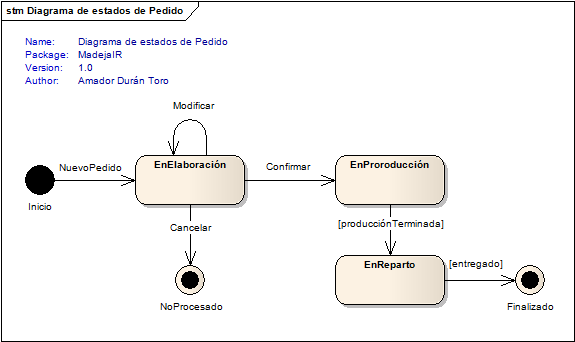


1. **DIAGRAMA DE ESTADO**

Es el diagrama que muestra a una máquina de estado, y son útiles para modelar la vida de un objeto

Es el que muestra el flujo de control entre estados (en que estados posibles puede estar algo y como se producen los cambios entre estos estados)

Una máquina de estado es un comportamiento que especifica las secuencias de estados por las que pasa un objeto a los largo de su vida en respuesta a eventos, junto con sus respuestas a esos eventos.



**CONCEPTOS CLAVES**

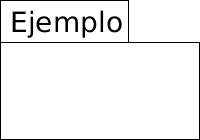
* ESTADO: un estado es una condición o situación en la vida de un objeto durante la cual satisface una condición, realiza alguna actividad o espera algún evento.
* EVENTO: es la especificación de un acontecimiento significativo que ocupa un lugar en el tiempo y en el espacio. Es la aparición de un estímulo que puede o no activar una transición de estado.
* TRANSICION: es una relación entre dos estados que indica que un objeto que está en el primer estado, realizará ciertas acciones y entrará en el segundo estado cuando ocurra un evento especificado y se satisfagan unas condiciones específicas.

Elementos:

* Círculo relleno, señalando al estado inicial.
* Círculo hueco que contiene un círculo más pequeño lleno, lo que indica el estado final (si existe)
* Rectángulo redondeado, lo que denota un estado. Parte superior del rectángulo contiene un nombre del estado. Puede contener una línea horizontal en el medio, por debajo del cual las actividades que se realizan en ese estado se indican
* Flecha, que denota la transición. El nombre del evento (si los hay) que causa esta transición etiqueta el cuerpo de la flecha. Una expresión de guardia puede añadirse antes un "/" y encerrado en corchetes cuadrados (eventName [guardExpression]), denotando que esta expresión debe ser cierto para la transición se lleve a cabo. Si se realiza una acción durante esta transición, que se añade a la siguiente etiqueta un "/" (eventName [guardExpression] / acción).
* Línea horizontal gruesa, ya sea con x> 1 líneas de entrada y 1 línea de salida o 1 línea de entrada y x> 1 líneas de salir. Estos denotan unirse / tenedor, respectivamente.

1. **DIAGRAMA DE PAQUETES**

Básicamente agrupan diferentes elementos de modelado en una entidad cohesiva. Estos por lo general se representan como una carpeta con el nombre de la misma en la pestaña.



Existen dos formas de escribir el nombre de lo que contiene el paquete, el simple, donde solo se coloca el nombre y el calificado, que muestra el nombre del paquete de la siguiente manera: Paquete::clase, donde "Paquete" el nombre del paquete que lo contiene y clase el elemento de modelado que contiene el paquete.

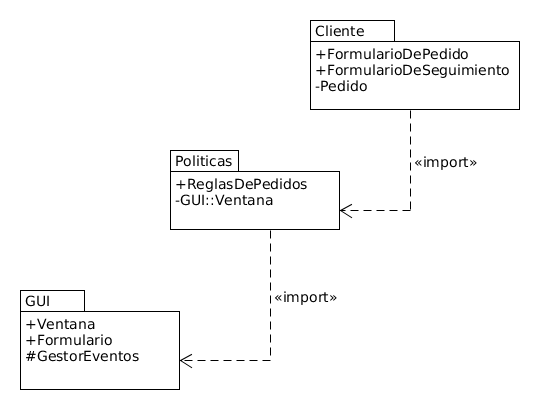
Cada paquete crea su propio espacio de nombre, es decir que puede haber dos o mas nombres iguales siempre que no estén en el mismo paquete; elementos de diferentes tipos pueden tener el mismo nombre pero no se recomienda.

Los paquetes pueden tener elementos tales como: clases, interfaces, componentes, nodos, colaboraciones, casos de uso, diagramas e incluso otro paquetes. Pero no es recomendable crear anidamientos muy profundos, como 2 o cuatro son suficientes

Al igual que las clases, se puede especificar la visibilidad de los elementos de un paquete. La visibilidad puede ser público, o privado colocando un signo "+", "-" o "#" a la izquierda respectivamente.

También existe el concepto de importación, que permite que si el paquete A importa al paquete B entonces el espacio de nombres de B se incluyen en A.

La importación se representa como una flecha punteada con la palabra <<import>> cerca de ella:



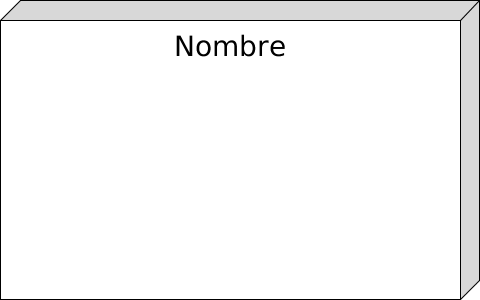
Elementos:

* Paquete: un mecanismo de propósito general para la organización de elementos y diagramas de modelos en grupos. Proporciona un espacio de nombres encapsulado dentro del cual todos los nombres deben ser únicos.
* Clase: una representación de un objeto que refleja su estructura y comportamiento dentro del sistema.
* Interfaz: una especificación de comportamiento. Una clase de implementación debe ser escrita para apoyar el comportamiento de una clase de interfaz.
* Objeto: una instancia de una clase. A menudo se utiliza en el análisis para representar un artefacto u otro elemento.
* Tabla: una clase estereotipada.

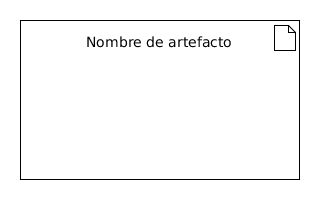
1. **DIAGRAMA DE DESPLIEGUE**

Los diagramas de despliegue modelan la interacción de piezas de hardware con elementos de software.

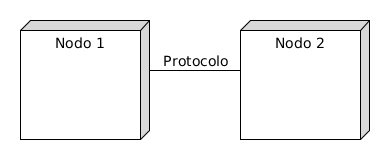
En estos diagramas existen "Nodos", que representan un elemento de hardware. Estos pueden ser monitores, procesadores, discos duros, teclados, etc. y se representan como un cubo con el nombre en la parte de arriba:



La parte del software también se puede representar como artefactos y son aquellos elementos de software que los nodos utilizan para sus funcionamientos. Para denotar en que nodo se encuentran desplegados se colocan dentro del diagrama de los nodos. Estos se representan gráficamente como un recuadro con el nombre en la parte de arriba y un símbolo de papel en la esquina superior izquierda:



También se puede especificar conexión entre nodos para denotar interacción entre ellos. Estos pueden representar enlaces físicos como conexión de red o bus de datos. Se representan con una línea simple con el nombre de la conexión o protocolo al lado



Elementos:

* Nodos (representados como un prisma)
* Componentes o Dispositivos (representados como una caja rectangular con dos protuberancias del lado izquierdo) y asociaciones.

1. **DIAGRAMA DE ACTIVIDAD**

Los diagramas de actividad modelan los aspectos dinámicos de un proceso y especifican en que orden ocurren los hechos. Estos diagramas son muy parecidos a los diagramas de flujos pero con la capacidad de especificar otros comportamientos como concurrencia.

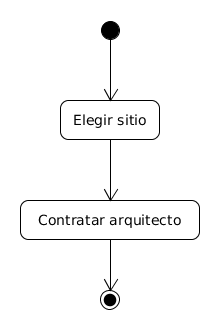
Estos contienen acciones, nodos de actividad, flujos. objetos valor:

Acciones: estos representan un paso que va a ser ejecutado en el proceso que se quiere modelar y se representa como un recuadro redondeado:

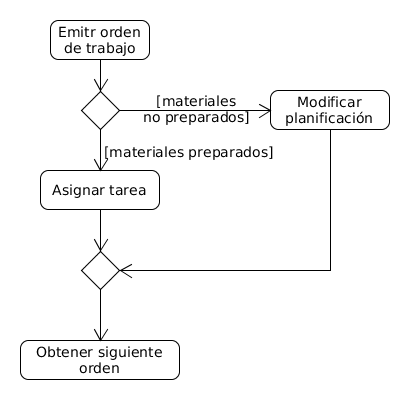


Nodos de actividad: Estos son elementos que encapsulan una serie de pasos en uno solo. A diferencia de las acciones, estos pueden ser divididos en otras acciones, es decir no son atómicos y se representan igual a las acciones excepto con paréntesis al final.

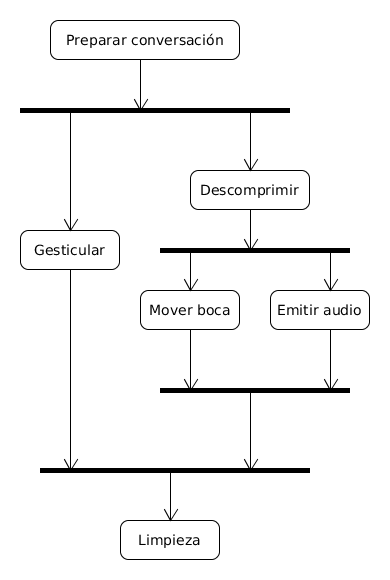
Flujos: Estos permiten describir el flujo u orden en que se ejecutan las acciones en el diagrama de actividad. Se describen con unas flechas y especifican cual acción ocurre después de otra. Para especificar el comienzo del proceso a modelar se coloca un punto relleno y se coloca una flecha saliendo de ella hacia la primera acción, y para especificar el fin se coloca un círculo con un círculo relleno dentro:



En estos diagramas se pueden tener bifurcaciones, es decir, puntos donde pueden haber flujos alternativos por recorrer. Las bifurcaciones se representan con un rombo y se debe colocar las condiciones de cada bifurcación cerca de las flechas de flujo. También se puede utilizar rombos como punto de unión de diferentes flujos:



Otra herramienta útil es la división y unión. La división representan momentos en donde dos o más flujos se ejecutan concurrentemente y la unión representa cuando se deben volver a sincronizar los flujos. Estos se representan con una barra vertical o horizontal.



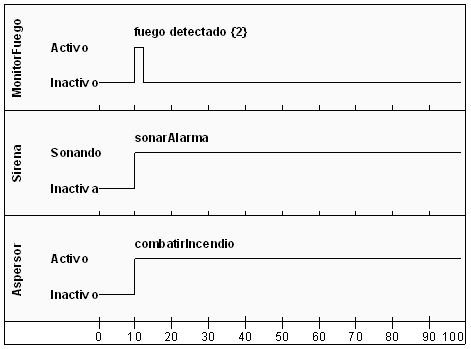
Flujos de objetos: especifican creación y manipulación de objetos en el diagrama de actividad. Estos se representan con recuadros, dentro se debe especificar el nombre del objeto y el estado entre corchetes.

Elementos del diagrama de actividades:

* **Círculo**: Procedimiento estandarizado.
* **Cuadrado**: Proceso de control.
* **Línea continua**: Flujo de información vía formulario o documentación en soporte de papel escrito.
* **Línea interrumpida**: Flujo de información vía formulario digital.
* **Rectángulo**: Formulario o documentación. Se grafica con un doble de ancho que su altura.
* **Rectángulo Pequeño**: Valor o medio de pago (cheque, pagaré, etc.). Se grafica con un cuádruple de ancho que su altura, siendo su ancho igual al de los formularios.
* **Triángulo (base inferior)**: Archivo definitivo.
* **Triángulo Invertido (base superior)**: Archivo Transitorio.
* **Semióvalo**: Demora.
* **Rombo**: División entre opciones.
* **Trapezoide**: Carga de datos al sistema.
* **Elipsoide**: Acceso por pantalla.
* **Hexágono**: Proceso no representado.
* **Pentágono**: Conector.
* **Cruz de Diagonales**: Destrucción de Formularios.

1. **DIAGRAMA DE TIEMPO:**

Es un diagrama de interacción que muestra los tiempos reales entre diferentes objetos o roles, en oposición a la simple secuencia relativa de mensajes. En el estándar de Lenguaje de Modelado Unificado de OMG los diagramas de tiempo son una representación especial de interacción que se enfoca en el tiempo de los mensajes enviados entre objetos.



Elementos:

* Objetos: Instancias de las clases y sus comportamientos en el tiempo.
* Comportamiento: El determinado comportamiento de un objeto en un tiempo dado.

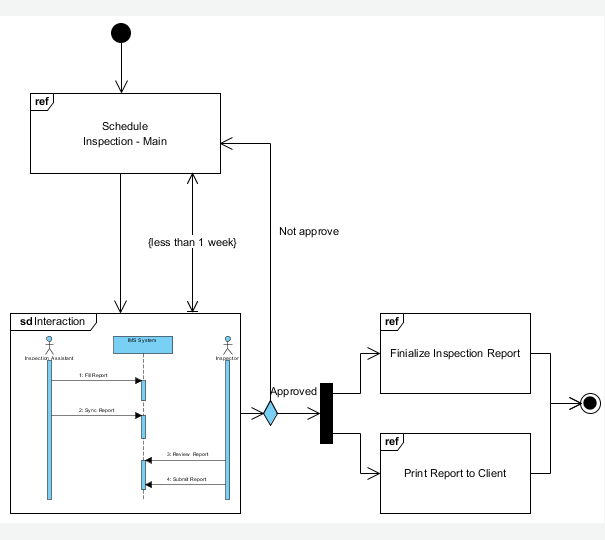
1. **DIAGRAMA DE VISIÓN GLOBAL DE INTERACCIONES:**

El diagrama global de las interacciones es un *diagrama de comportamiento*, más precisamente, uno de los cuatro *diagramas de interacción*. Muestra una cierta vista sobre los aspectos dinámicos de los sistemas modelados.

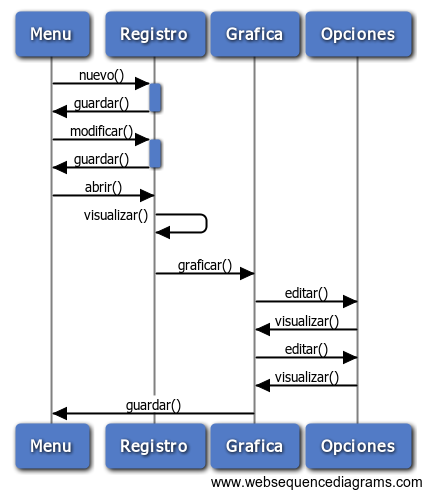
Es un hibrido entre un diagrama de actividades y un diagrama de secuencia

Elementos:

* Círculo relleno, señalando al estado inicial.
* Círculo hueco que contiene un círculo más pequeño lleno, lo que indica el estado final (si existe).



**Diagrama de secuencia:** es un tipo de diagrama usado para modelar interacción entre objetos en un sistema.



Objetos usados en el diagrama de secuencia:

* Mensaje: pasa de la línea de vida de un objeto a otro.
* Objeto: rectángulos con nombres subrayados, el tiempo se Representa como una progresión vertical.
* Línea de vida activa: el tiempo se representa en forma vertical inicia en la parte superior y avanza a la parte inferior un mensaje que este en la parte superior ocurrirá antes que uno en la parte inferior.

1. **CONCLUSIONES**:

Luis Correa

* En el diagrama de tiempo, un sistema secundario se activa como consecuencia de la activación de un sistema primario.

Ejemplo: El sistema secundario (Alarma y Aspersores) se activa una vez que se activa el sistema primario (Detector de Humo).

* En el diagrama de la visión global de interacciones: Tendremos un diagrama de secuencia de ofimática y otro de un software de almacenamiento en la nube (internet).

Para almacenar el software de ofimática en la nube se tendrá que cumplir con la condición de que haya iniciado sesión en el software de almacenamiento en la nube y luego se podrá enviar el archivo del diagrama de ofimática al diagrama de almacenamiento en la nube.

Orlando Medina

* Es fácil predecir que UML será el lenguaje de modelado de software de uso universal. Esto debido a que la mayoría de las empresas importantes de informática la han apoyado, y la han aceptado como un estándar.
* Con respecto a los diagramas, no es una actividad de dibujar grafos. Se trata de escribir con el detalle necesario, el flujo principal de un sistema y los flujos alternativos.
* El objetivo esencial es identificar los actores involucrados y a partir de sus objetivos, encontrar los casos de uso. Los diagramas de comunicación en especial, son una ayuda visual del comportamiento de un sistema.

Manuel Dun

* Los diagramas de paquetes son herramientas poderosas en proyectos de gran escala, logran simplificar el desarrollo de manera eficaz y son fáciles de entender. Además de esto, son simple de implementar y ayudan a disminuir la complejidad de proyectos de desarrollo de software.
* Los diagramas de despliegue se utilizan para documentar cómo el hardware y el software interactúan en el proyecto, especifican los diferentes elementos que se involucran y cómo se relacionan entre ellos.
* Los diagramas de actividad ayudan a modelar los comportamientos involucrados en algún proceso del proyecto a desarrollar. Estos diagramas son muy parecidos a los diagramas de flujos que todos conocemos, añadido a esto, se pueden utilizar tanto para modelar algoritmos específicos a implementar en un lenguaje de programación, como para contextos m[as generales que talvez no se vayan a implementar como un programa para ejecutarse.