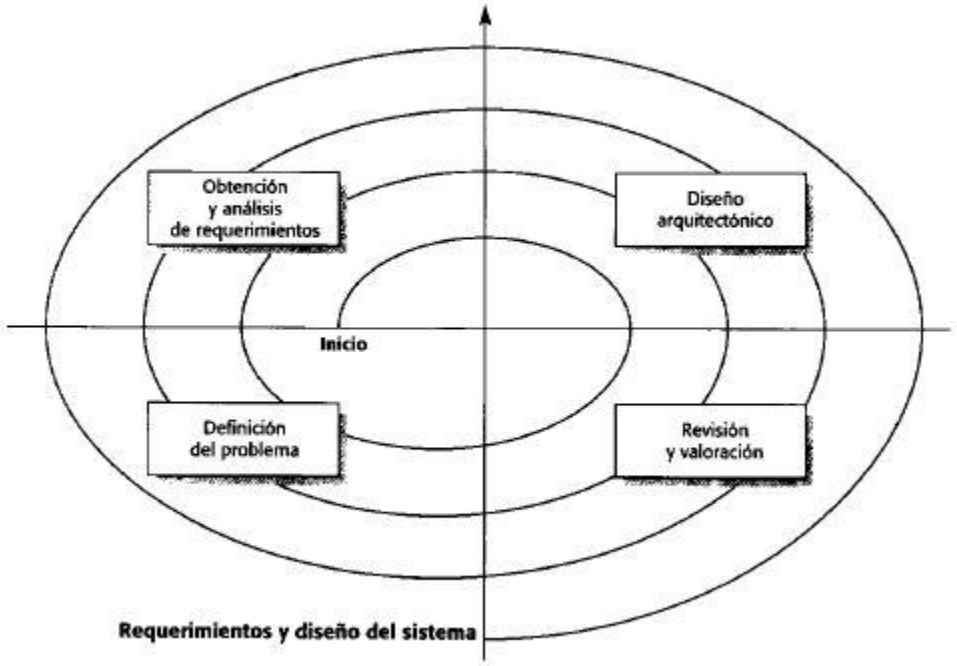
Resumen.

3. Diseño del software.



Proceso de desarrollo de software en espiral

La etapa de implementación en el desarrollo de software es el proceso de convertir una especificación de un sistema en un ejecutable. Siempre requiere de procesos de diseño de software y programación, pero si se aplica una aproximación evolucionaria al desarrollo también puede incluir refinamiento de las especificaciones del software.

Un diseño de software es una descripción de la estructura del software a ser implementado, los datos que forman parte del sistema, las interfaces de los componentes del sistema y, algunas veces, los algoritmos usados.

Los diseñadores no llegan inmediatamente a un diseño terminado sino que este es desarrollado iterativamente a través de un número de versiones.

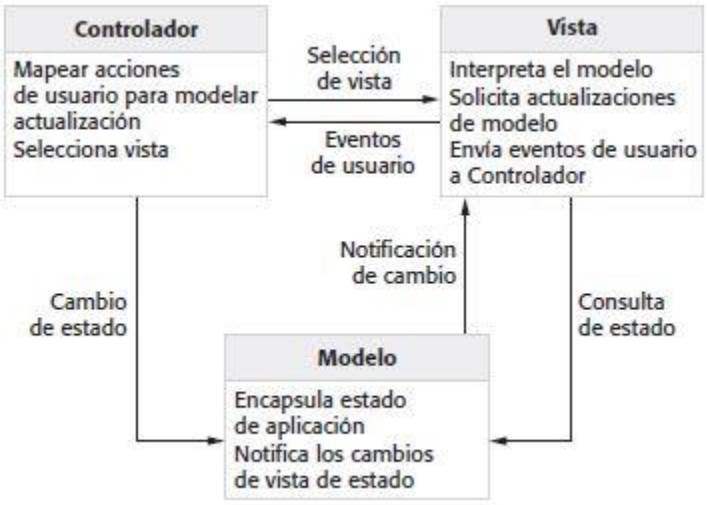
El proceso de diseño requiere agregar formalidad y detalle conforme el diseño es desarrollado, con un constante objetivo de corregir diseños anteriores.

El proceso de diseño también puede incluir y desarrollar diversos modelos del sistema en niveles diferentes de abstracción. Conforme se descompone un diseño, errores y omisiones en etapas anteriores se develan.

La retroalimentación de una etapa a la otra y el consiguiente retrabajo del diseño ambos son inevitables en cualquier proceso de diseño.

Las actividades específicas dentro del proceso de diseño son:

1. Diseño de arquitectura: Los subsistemas que conforman el sistema y sus relaciones son identificadas y documentadas.
2. Especificación abstracta: Para cada subsistema, una especificación abstracta de los servicios y limitaciones bajo la cual deben operar es producida.
3. Diseño de interfaces: Para cada subsistema, su interface con otros subsistemas es diseñada y documentada. La especificación de la interface debe ser no ambigua ya que permite que el subsistema sea usado sin conocimiento previo de su operación.
4. Diseño de componentes: Los servicios son alojados en diferentes componentes y la interface de estos componentes es diseñada
5. Diseño de estructuras de datos: Las estructuras de datos usadas en la implementación del sistema son diseñadas en detalle y especificadas.
6. Diseño de algoritmos: Los algoritmos usados para proveer servicios son diseñados en detalle y especificados.



Modelo vista controlador, un ejemplo de diseño de arquitectura

3.1 Fundamentos para el diseño de software

Diseño arquitectónico.

El diseño arquitectónico se interesa por entender cómo debe organizarse un sistema y cómo tiene que diseñarse la estructura global de ese sistema. En el modelo del proceso de desarrollo de software, como se mostró en el capítulo 2, el diseño arquitectónico es la primera etapa en el proceso de diseño del software. Es el enlace crucial entre el diseño y la ingeniería de requerimientos, ya que identifica los

principales componentes estructurales en un sistema y la relación entre ellos. La salida del proceso de diseño arquitectónico consiste en un modelo arquitectónico que describe la forma en que se organiza el sistema como un conjunto de componentes en comunicación.

Las arquitecturas de software se diseñan en dos niveles de abstracción, arquitectura en pequeño y arquitectura en grande:

1. La arquitectura en pequeño se interesa por la arquitectura de programas individuales. En este nivel, uno se preocupa por la forma en que el programa individual se separa en componentes. Este capítulo se centra principalmente en arquitecturas de programa.

2. La arquitectura en grande se interesa por la arquitectura de sistemas empresariales complejos que incluyen otros sistemas, programas y componentes de programa. Tales sistemas empresariales se distribuyen a través de diferentes computadoras, que diferentes compañías administran y poseen. En los capítulos 18 y 19 se cubren las arquitecturas grandes; en ellos se estudiarán las arquitecturas de los sistemas distribuidos.

Ventajas de diseñar y documentar de manera explícita la arquitectura de software:

1. Comunicación con los participantes La arquitectura es una presentación de alto nivel del sistema, que puede usarse como un enfoque para la discusión de un amplio número de participantes.

2. Análisis del sistema En una etapa temprana en el desarrollo del sistema, aclarar la arquitectura del sistema requiere cierto análisis. Las decisiones de diseño arquitectónico tienen un efecto profundo sobre si el sistema puede o no cubrir requerimientos críticos como rendimiento, fiabilidad y mantenibilidad.

3. Reutilización a gran escala Un modelo de una arquitectura de sistema es una descripción corta y manejable de cómo se organiza un sistema y cómo interoperan sus componentes. Por lo general, la arquitectura del sistema es la misma para sistemas con requerimientos similares y, por lo tanto, puede soportar reutilización de software a gran escala. Como se explica en el capítulo 16, es posible desarrollar arquitecturas de línea de productos donde la misma arquitectura se reutilice mediante una amplia gama de sistemas relacionados.

Diseño orientado a objetos con el uso del UML

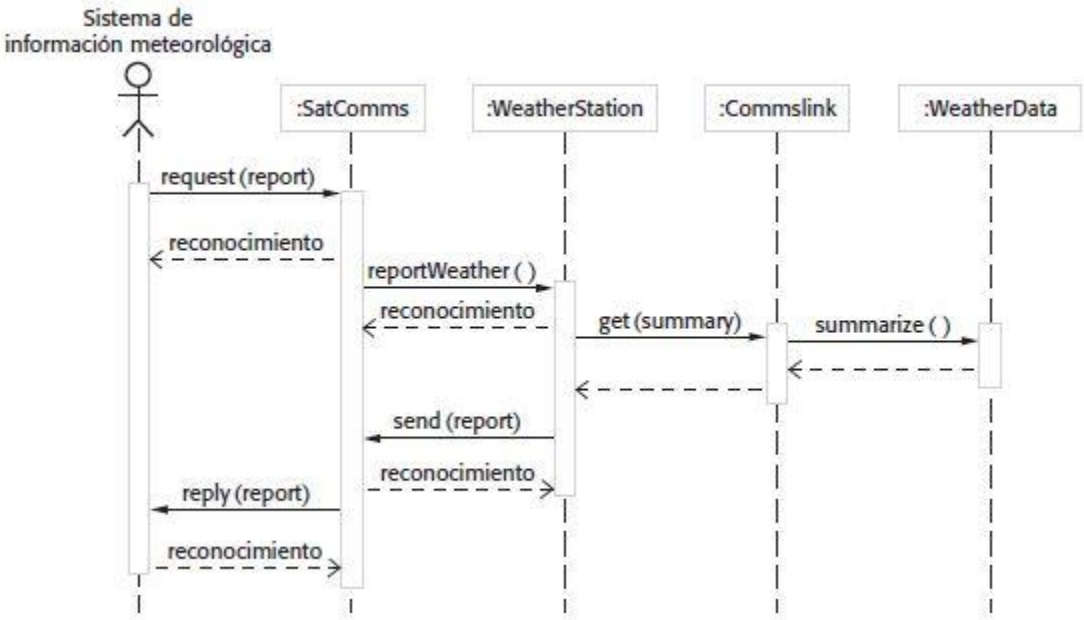
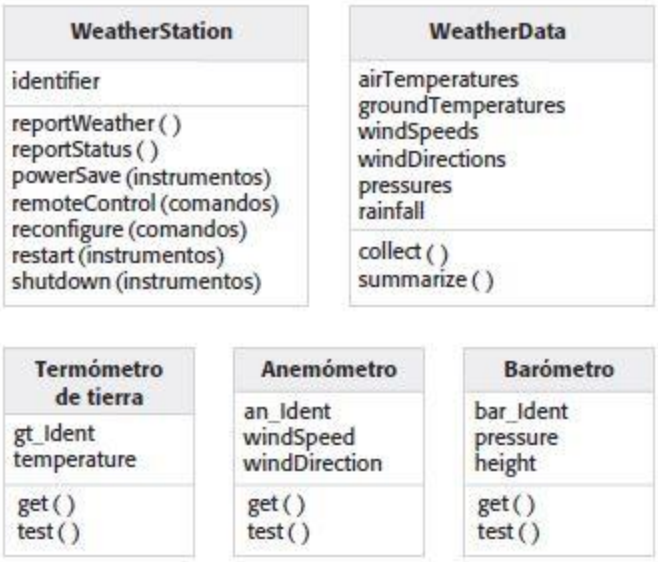


Diagrama de secuencia.

Un sistema orientado a objetos se constituye con objetos que interactúan y mantienen su propio estado local y ofrecen operaciones sobre dicho estado. La representación del estado es privada y no se puede acceder directamente desde afuera del objeto. Los procesos de diseño orientado a objetos implican el diseño de clases de objetos y las relaciones entre dichas clases; tales clases definen tanto los objetos en el sistema como sus interacciones.

Cuando el diseño se realiza como un programa en ejecución, los objetos se crean dinámicamente a partir de estas definiciones de clase.



Los sistemas orientados a objetos son más fáciles de cambiar que aquellos sistemas desarrollados usando enfoques funcionales. Los objetos incluyen datos y operaciones para manipular dichos datos. En consecuencia, pueden entenderse y modificarse como entidades independientes. Cambiar la implementación de un objeto o agregar servicios no afectará a otros objetos del sistema. Puesto que los objetos se asocian con cosas, con frecuencia hay un mapeo claro entre entidades del mundo real (como componentes de hardware) y sus objetos controladores en el sistema. Esto mejora la comprensibilidad y, por ende, la mantenibilidad del diseño.

Para desarrollar un diseño de sistema desde el concepto hasta el diseño detallado orientado a objetos, hay muchas cuestiones por hacer:

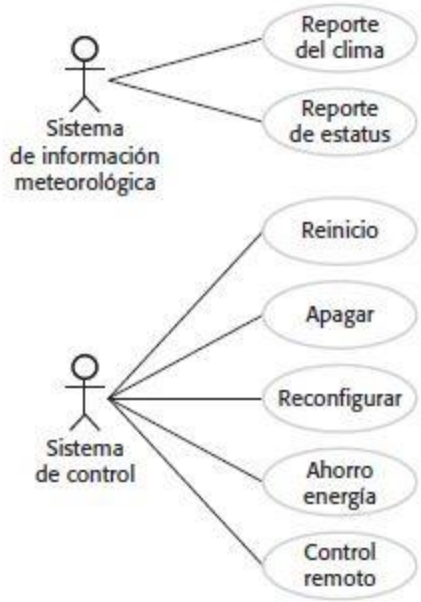
1. Comprender y definir el contexto y las interacciones externas con el sistema.

2. Diseñar la arquitectura del sistema.

3. Identificar los objetos principales en el sistema.

4. Desarrollar modelos de diseño.

5. Especificar interfaces.



Diseño arquitectónico

Una vez definidas las interacciones entre el sistema de software y el entorno del sistema, se aplica esta información como base para diseñar la arquitectura del sistema

3.1.1 Conceptos generales del diseño.

Ingeniería de sistemas.

La ingeniería de sistemas es la actividad de especificar diseñar, implementar, validar, utilizar y mantener los sistemas socio-técnicos. Los ingenieros de sistemas no sólo tratan con el software, sino también con el hardware y las interacciones del sistema con los usuarios y su entorno. Deben pensar en los servicios que el sistema proporciona, las restricciones sobre las que el sistema se debe de construir y funcionar y las formas en las que el sistema es usado para cumplir con su propósito.

En la siguiente imagen se muestran las fases del proceso de ingeniería de sistemas, este modelo tiene una gran influencia sobre el modelo en cascada.

Definición de requerimientos del sistema.

Al definir requerimientos del sistema se especifica que es lo que el sistema debe hacer (sus funciones) y sus propiedades esenciales y deseables. Crear una definición de requerimientos del sistema requiere de consultar con los clientes del sistema y los usuarios finales.

Al definir requerimientos se identifican tres tipos:

1. Requerimientos funcionales abstractos:

   Las funciones básicas que el sistema debe proporcionar se definen en un nivel abstracto. Una especificación más detallada de requerimientos funcionales tiene lugar en el nivel de subsistemas.

2. Propiedades del sistema:

   Son propiedades emergentes no funcionales del sistema, tales como disponibilidad, rendimiento y seguridad. Estas propiedades no funcionales del sistema afectan los requerimientos de todos los subsistemas.

3. Características que no debe mostrar el sistema:

   Algunas veces es tan importante especificar lo que el sistema no debe hacer como lo que debe hacer.

Diseño del sistema.

El diseño del sistema se centra en proporcionar la funcionalidad del sistema a través de sus diferentes componentes. Las actividades realizadas en este proceso son:

1. Dividir requerimientos: Analice los requerimientos y organícelos en grupos afines. Normalmente existen varias opciones posibles de división, y puede surgir varias alternativas en esta etapa del proceso.
2. Identificar subsistemas: Debe identificar los diferentes subsistemas que pueden, individual o colectivamente, cumplir los requerimientos. Los grupos de requerimientos están normalmente relacionados con los subsistemas, de tal forma que esta actividad y la división de requerimientos se pueden fusionar.
3. Asignar requerimientos a los subsistemas: Asigne los requerimientos a los subsistemas. En principio, esto debe ser sencillo si la división de requerimientos se utiliza para la identificación de subsistemas.
4. Especificar la funcionalidad de los subsistemas: Debe enumerar las funciones específicas asignadas a cada subsistema. Esto puede verse como parte de la fase de diseño del sistema o, si el subsistema es un sistema de software, como parte de la actividad de especificación de requerimientos para ese sistema
5. Definir las interfaces del susbsistema: Defina las interfaces necesarias requeridas por cada subsistema.

3.1.2 El contexto del diseño de software.

Métodos de diseño.

En muchos proyectos de desarrollo, el diseño de software es un proceso adhoc; empezando por un conjunto de requerimientos, usualmente dados en lenguaje natural, se prepara un diseño informal.
Cuando la programación en si comienza el diseño se modifica conforme el sistema es implementado, hay poco o ningún control formal de los cambios o administración del diseño. Por ello cuando la etapa de implementación está completa, el diseño ha cambiado tanto con respecto a su especificación inicial, que el documento original del diseño es una descripción incorrecta o incompleta del sistema.
Con la introducción de los métodos estructurados, una aproximación más metódica al diseño de software se logra, estos métodos estructurados son en realidad un conjunto de notaciones y guías para el diseño de software.

Un método estructurado incluye un modelo de proceso de diseño, notaciones para representar al diseño, formatos de reporte, y guías de reglas y diseño.
Los métodos estructurados pueden soportar algunos o todos los siguientes modelos de un sistema:

1. Un modelo de flujo de datos donde el sistema es modelado usando las transformaciones de datos que se llevan a cabo conforme son procesados.

2. Un modelo entidad-relación que se usa para describir la entidades básicas en el diseño, y las relaciones entre estos. Los modelos entidad relación son comunes para representar la estructura de una base de datos.

3. Un modelo estructurado donde los componentes del sistema y sus interacciones se documentan.

4. Métodos orientados a objetos que incluyen el modelo de herencia del sistema, asi como modelos de las relaciones estáticas y dinámicas entre los objetos y un modelo de como los objetos interaccionan entre ellos cuando el sistema se está ejecutando.

Figura 1. Representación gráfica de una clase

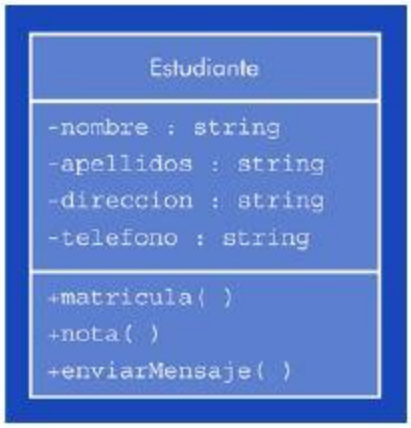


Diagrama de clases, un ejemplo típico de diseño orientado a objetos