Diseño orientado a objetos

Diseñar sistema usando objetos autocontenidos y clases de objetos

Objetivos

- Explicar como un diseño de software puede ser representado como un conjunto de objetos que interactúan manejando su propio estado y operaciones
- Describir las actividades en el proceso de diseño orientado a objetos
- Introducir varios modelos que describen un diseño orientado a objetos
- Mostrar como el UML puede se usado para representar estos modelos

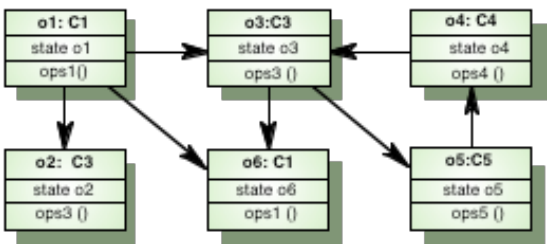
Tópicos cubiertos

- Objetos y clases de objetos
- Un proceso de diseño orientado a objetos
- Evolución del diseño

Características del DOO

- Los objetos son abstracciones del mundo real o entidades del sistemas y manejarse ellos mismos
- Los objetos son independientes y encapsulan estado y representan información
- La funcionalidad del sistema se expresa en términos de servicios de objetos
- Las áreas de datos compartidos se eliminan. Los objetos se comunican a través de paso de mensajes
- Los objetos pueden se distribuidos y pueden se ejecutados secuencialmente o en paralelo

Objetos interactuando



Ventajas de el DOO

- Mantenimiento más fácil. Los objetos se pueden entender como entidades independientes
- Los objetos son componentes apropiados reusables
- Por algunos sistemas, puede haber una obvia correspondencia entre entidades del mundo real y objetos del sistema

Desarrollo orientado a objetos

- El análisis, diseño y programación orientados a objetos están relacionados pero distintos
- El AOO es concerniente con el desarrollo de un modelo de objetos del dominio de la aplicación
- El DOO es concerniente con el desarrollo de un sistema orientado a objetos del sistema para implementar requerimientos
- La POO es concerniente con la implementación de un DOO usando un lenguaje OO tal como java o C++

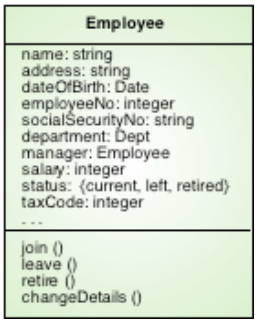
Objetos y clases

- Los objetos son entidades en un sistema de software los cuales representan instancias del mundo real y entidades del sistema
- Las clases de objetos son plantilla de objetos las cuales pueden ser usadas para crearlos
- Las clases de objetos pueden heredar atributos y servicios de otras clases de objetos

Objetos

Un objeto es una entidad la cual tiene un estado y un conjunto definido de operaciones las cuales modifican el estado. El estado se representa como un conjunto de atributos del objeto. Las operaciones asociadas con el objeto proveen servicios a otros objetos (clientes) los cuales solicitan estos servicios cuando alguna computación se requiere.
Los objetos son creados de acuerdo a algunas definiciones de clases de objetos. Una definición de clase sirve como una plantilla para objetos. Incluye declaraciones de todos los atributos y servicios que deben ser asociados con un objeto de esa clase.

Employee object class (UML)



Comunicación de objetos

- Conceptualmente, los objetos se comunican a través del paso de mensajes
- Mensajes
  - El nombre del servicio requerido por el objeto llamante
  - Copias de la información requerida para ejecutar el servicio y el nombre de un receptor para el resultado del servicio
- En la práctica, los mensajes son a menudo implementado por llamadas de procedimientos
  - Nombre = nombre del procedimiento
  - Información = lista de parámetros

Ejemplo de un mensaje

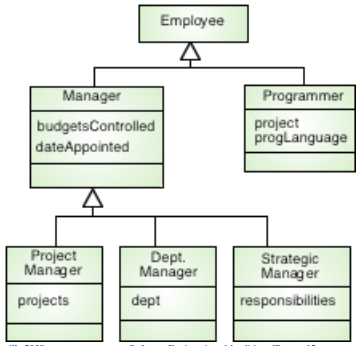
```
// Call a method associated with a buffer
// object that returns the next value
// in the buffer
        v = circularBuffer.Get () ;

// Call the method associated with a
// thermostat object that sets the
// temperature to be maintained
        thermostat.setTemp (20) ;
```

Herencia y generalización

- Los objetos son miembros de clases las cuales definen atributos y operaciones
- La clases pueden se organizadas en una jerarquía de clases donde una clase (superclase) es una generalización de un o más clases adicionales (subclases)
- Una subclase hereda los atributos y operaciones de su superclase y pueden ser adicionados nuevos métodos o atributos propios
- La generalización en UML se implementa como herencia en lenguajes de POO

Jerarquía de generalización



Ventajas de la herencia

- Mecanismo de abstracción que puede ser usado para clasificar entidades
- Mecanismo de reutilización tanto a nivel de diseño y programación
- El diagrama de herencia es una fuente de conocimiento organizacional acerca del dominio y del sistema

Problemas con la herencia

- Las clases de objetos no son autocontenidas. No pueden ser entendidas sin referirse a las superclases
- Los diseñadores tienen una tendencia a reutilizar el diagrama de herencia creado durante el análisis. Esto puede ser muy ineficiente
- Los diagramas de herencia análisis, diseño e implementación tienen funciones diferentes y deben ser mantenidos por separado

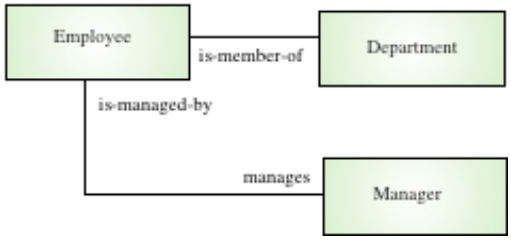
Herencia y DOO

- Hay diferentes puntos de vista respecto a la importancia de la herencia en DOO
  - Identificar la jerarquía de herencia es una parte fundamental de DOO. Obviamente esto puede ser solamente implementado usando un LPOO
  - La herencia es un concepto de implementación útil el cual permite reutilizar definiciones de atributos y operaciones. Identificar un jerarquía de de herencia en la etapa de diseño pone restricciones innecesarias en la implementación
- La herencia introduce complejidad y esto es indeseable, especialmente en sistemas críticos

Asociaciones en UML

- Objetos y clases participan en relaciones con otros objetos y clases
- En UML, una relación generalizada se indica por una asociación
- Las asociaciones pueden se anotadas con información que describe la asociación
- Las asociaciones son generales pero pueden indicar que un atributo de un objeto es un objeto asociado o que un método depende de un objeto asociado

Un modelo de asociación



Objetos concurrentes

- La naturaleza de los objetos como entidades autocontenidas las hace adecuadas para implementación concurrente
- El modelo de paso de mensajes para la comunicación de objetos puede se implementado directamente si los objetos corren en procesadores separados en un sistema distribuido

Servidores y objetos activos

- Servidores
  - El objeto es implementado como un proceso paralelo con puntos de entrada correspondientes a operaciones del objeto. Si no se hacen llamadas, el objeto se suspende por si mismo y espera por solicitudes de servicio
- Objetos activos
  - Los objetos son implementados como procesos paralelos y el estado interno del objeto puede ser cambiado por el mismo objeto y no solamente con llamadas externas.

Active transponder object

- Active objects may have their attributes modified by operations but may also update them autonomously using internal operations
- Transponder object broadcasts an aircraft's position. The position may be updated using a satellite positioning system. The object periodically update the position by triangulation from satellites

An active transponder object

```
class Transponder extends Thread {

        Position currentPosition ;
        Coords c1, c2 ;
        Satellite sat1, sat2 ;
        Navigator theNavigator ;

        public Position givePosition ()
        {
                return currentPosition ;
        }

        public void run ()
        {
                while (true)
                {
                        c1 = sat1.position () ;
                        c2 = sat2.position () ;
                        currentPosition = theNavigator.compute (c1, c2) ;
                }

        }

} //Transponder
```


Hilos en Java

- Los hilos en Java son un mecanismo simple para la implementación de objetos concurrentes
- Los hilos pueden incluir un método llamado run() y este es arrancado por el sistema de tiempo de ejecución de Java
- Los objetos activos incluyen típicamente un ciclo infinito de manera que siempre están llevando a cabo el cómputo

Un proceso de diseño orientado a objetos

- Defina el contexto y los modos de uso del sistema
- Diseñe la arquitectura del sistema
- Identifique los objetos principales del sistema
- Desarrolle modelos de diseño
- Especifique las interfaces de objetos

Sistema de descripción del clima

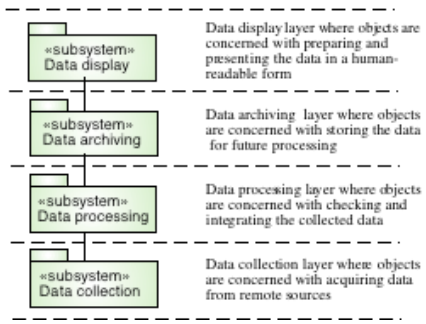
Un sistema de recolección de información climática debe generar mapas climáticos de manera rutinaria usando información tomada de estaciones climáticas remotas y desatendidas, tales como observadores, globos y satélites climáticos. Las estaciones climáticas transmiten sus datos al computador del área como respuesta al requerimiento desde esa máquina.

El computador del área valida la información recolectada y la integra con la información de diferentes fuentes. La información integrada es archivada y, usando datos de este archivo y una BD de mapas digitalizados, un conjunto de de mapas climáticos locales es creado. Los mapas pueden ser impresos para distribución en una impresora de mapas de propósito especial o puede ser desplegado en un número de formatos diferentes.

Descripción de la estación climática

Una estación climática es un paquete de instrumentos controlados por software los cuales recolectan datos, desempeñan procesamiento de datos y transmiten estos datos para procesamiento adicional. Los instrumentos incluyen termómetros aéreos y de tierra, un anemómetro, un veleta, un barómetro y un medidor de lluvia. Los datos se recolectan cada cinco minutos. Cuando un comando se genera para transmitir los datos climáticos, la estación climática procesa y resume los datos recolectados. Los datos resumidos se transmiten al computador que procesa el mapa cuando un requerimiento es recibido.

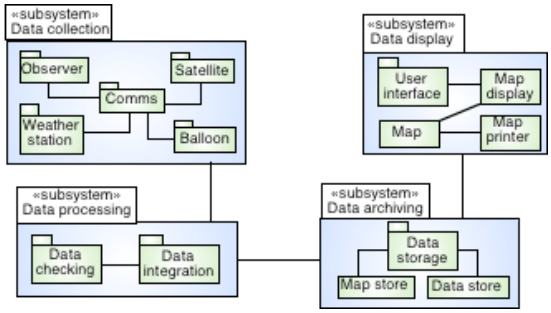
Arquitectura en capas



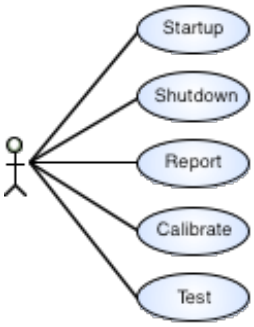
Contexto del sistema y modelos de uso

- Desarrollar un entendimiento de las relaciones entre el software a siendo diseñado y su ambiente externo
- Contexto del sistema
  - Un modelo estático que describe otros sistemas en el ambiente. Use un modelo de subsistemas para mostrar otros sistemas. La siguiente diapositiva muestra el sistema alrededor del sistema de estación climática
- Modelo de uso del sistema
  - Un modelo dinámico describe como el sistema interactúa con su ambiente. Use casos de uso para mostrar las interacciones

Subsistemas en el sistema de mapeo climático



Casos de uso para la estación climática



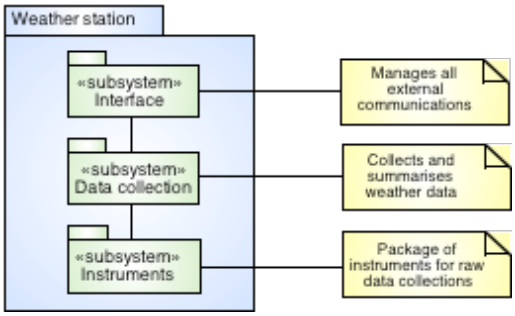
Descripción de caso de uso

System	Weather station
Use-case	Report
Actors	Weather data collection system, Weather station
Data	The weather station sends a summary of the weather data that has been collected from the instruments in the collection period to the weather data collection system. The data sent are the maximum minimum and average ground and air temperatures, the maximum, minimum and average air pressures, the maximum, minimum and average wind speeds, the total rainfall and the wind direction as sampled at 5 minute intervals.
Stimulus	The weather data collection system establishes a modem link with the weather station and requests transmission of the data.
Response	The summarised data is sent to the weather data collection system
Comments	Weather stations are usually asked to report once per hour but this frequency may differ from one station to the other and may be modified in future.

Diseño arquitectónico

- Una vez la interacciones entre el sistema y su ambiente ha sido entendido, usted usa esta información para diseñar la arquitectura del sistema
- La arquitectura por capas es apropiada para la estación climática
  - Capa de interfaz para manejar las comunicaciones
  - Capa de recolección de datos para manejar instrumentos
  - Capa de instrumentos para recolectar datos
- No deben haber mas de 7 entidades en un modelo arquitectónico

Weather station architecture



Identificación de objetos

- La identificación de objetos (o clases) es la parte mas difícil del DOO
- No existe una “fórmula mágica” para la identificación de objetos. Depende de la pericia, experiencia y conocimiento del dominio de los diseñadores del sistema
- La identificación de objetos es un proceso iterativo. Es muy improbable que usted lo haga perfectamente la primera vez

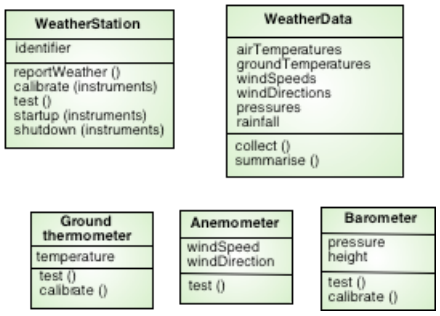
Estrategias de identificación

- Use una estrategia gramatical basada en una descripción en lenguaje natural del sistema
- Base la identificación en objetos tangibles en el dominio de la aplicación
- Use una estrategia basada en comportamiento e identifique objetos basado en que participa en que comportamiento
- Use un análisis basado en escenarios. Los objetos, atributos y métodos en cada escenario son identificados

Clases de la estación climática

- Termómetro terrestre, Anemómetro, Barómetro
  - Objetos del dominio de la aplicación que son objetos de 'hardware' relacionados a los instrumentos en el sistema
- Estación climática
  - La interfaz básica de la estación climática a su ambiente. Por lo tanto refleja las interacciones identificadas en el modelo de casos de uso
- Datos climáticos
  - Encapsula los datos encapsulados desde los instrumentos

Weather station object classes



Objetos adicionales y refinamiento de objetos

- Use conocimiento del dominio para identificar más objetos y operaciones
  - Las estaciones climáticas deben tener un identificador único
  - La estaciones climáticas están situadas remotamente, por lo tanto las fallas de los instrumentos deben ser reportadas automáticamente. De manera que se requieren atributos y operaciones para autochequeo
- Objetos activos o pasivos
  - En este caso, los objetos son pasivos y recolectan datos por requerimiento y no de manera autónoma. Esto introduce flexibilidad a expensas del tiempo de proceso del controlador

Modelos de diseño

- Los modelos de diseño muestran los objetos, clases y relaciones entre estas entidades
- Los modelos estáticos describen la estructura estática de el sistema en términos de clases y relaciones
- Los modelos dinámicos describen las interacciones dinámicas entre objetos

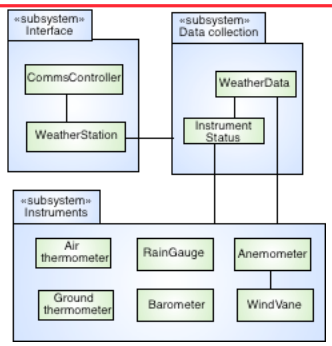
Ejemplos de modelos de diseño

- Modelos de subsistemas que muestran las agrupaciones lógicas de objetos en subsistemas coherentes
- Modelos de secuencia que muestran la secuencia de interacciones entre objetos
- Modelos de máquinas de estado que muestran cómo objetos individuales cambian su estado en respuesta a eventos
- Otros modelos incluyen modelos de agregación, modelos de generalización, etc.

Modelos de subsistemas

- Muestra como el diseño es organizado en grupos lógicos, relacionados de objetos
- En UML, estos modelos se muestran usando paquetes. Este es un modelo lógico. La organización real de objetos en el sistema pueden ser diferentes

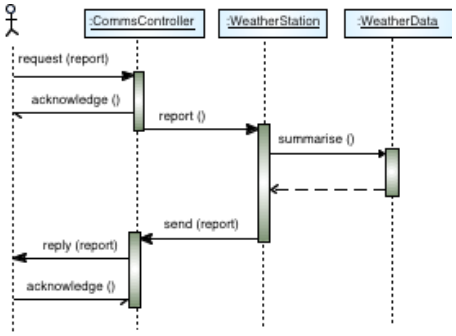
Subsistema de estación climática



Modelos de secuencia

- Los modelos de secuencia muestran la secuencia de interacción entre objetos
  - Los objetos son organizados horizontalmente en la parte superior
  - El tiempo se representa verticalmente, de manera que los modelos se leen de arriba hacia abajo
  - Las interacciones son representadas por flechas etiquetadas. Estilos diferentes de flechas, representan diferentes tipos de interacción
  - Un rectángulo delgado en la línea de vida de un objeto representa el tiempo cuando el objeto con el control en el sistema

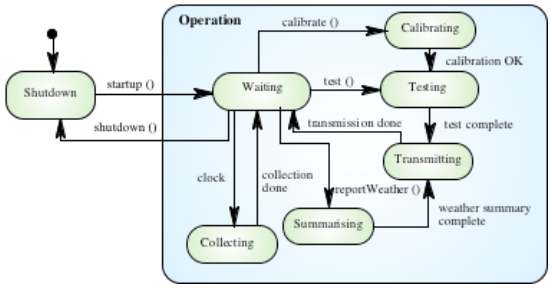
Secuencia de recolección de datos



Diagramas de estado

- Muestra cómo los objetos responden a diferentes requerimientos de servicios y la transición de estado generada por este requerimiento
  - Si el estado del objeto es *Shutdown* entonces él responde a un mensaje *Startup()*
  - Si el mensaje *reportWeather()* es recibido, el sistema cambia al estado *Summarising*
  - Si el mensaje *calibrate()* es recibido, el sistema se mueve al estado *Calibrating*
  - Se entra al estado *Collecting* cuando un signo del reloj se recibe

Diagrama de estados de la estación climática



Especificación de la interfaz de objeto

- Las interfaces de objetos tienen que ser especificados de tal forma que los objetos y otros componentes puedan ser diseñados en paralelo
- Los diseñadores deben evitar diseñar la representación de de la interfaz, en cambio debe ocultarla en el objeto como tal
- Los objetos pueden tener varias interfaces las cuales son puntos de vista de los métodos provistos
- El UML usa diagramas de clase para especificación de interfaces pero es posible usar Java

Interfaz de la estación climática

```
interface WeatherStation {

        public void WeatherStation () ;

        public void startup () ;
        public void startup (Instrument i) ;

        public void shutdown () ;
        public void shutdown (Instrument i) ;

        public void reportWeather ( ) ;

        public void test () ;
        public void test ( Instrument i ) ;

        public void calibrate ( Instrument i) ;

        public int getID () ;

} //WeatherStation
```

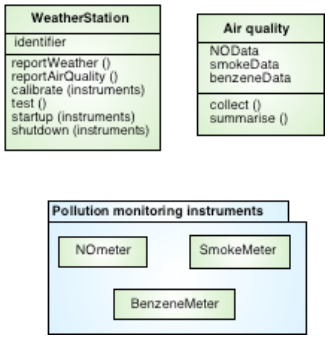

Evolución del diseño

- El ocultamiento de la información dentro de los objetos significa que los cambios hechos a un objeto no afectan otros objetos en una forma impredecible
- Suponga que se deben adicionar facilidades para monitorear polución a las estaciones climáticas. Estos muestrean el aire y computan la cantidad de diferente contaminantes en la atmósfera
- Las lecturas de polución son transmitidos con datos climáticos

Cambios requeridos

- Adiciones una clase llamada 'Air quality' como parte de WeatherStation
- Adicione una operación reportAirQuality a WeatherStation. Modifique el software de control para recolectar lecturas de polución
- Adicione objetos que representen instrumentos de monitoreo de polución

Monitoreo de polución



Puntos claves

- DOO es una estrategia de diseño tal que los componentes de diseño tienen su estado y operaciones privados propios
- Los objetos deben tener constructores y operaciones de inspección. Ellos proveen servicios a otros objetos
- Los objetos pueden ser implementados secuencialmente o concurrentemente
- UML provee notaciones diferentes para definir diferentes modelos de objetos

Puntos claves

- Un rango de modelos diferentes pueden se producidos durante un proceso de diseño orientado a objetos. Estos incluyen modelos de sistema estáticos y dinámicos
- Las interfaces de objetos deben ser definidos precisamente usando, por ejemplo, un lenguaje de programación como Java
- El DOO simplifica la evolución del sistema