Diseño de paquetes

Conforme el tamaño de las aplicaciones crece,
se hace necesario algún tipo de organización a alto nivel.

Las clases son unidades demasiado pequeñas,
por lo que se agrupan en paquetes.

¿Qué criterios utilizaremos para organizar las clases?

Los mismos que para organizar los miembros de las clases:

la **cohesión** y el **acoplamiento**

Granularidad: La cohesión de los paquetes

¿Cuándo se ponen dos clases en el mismo paquete?

- Cuando para usar una, siempre es **necesario** usar la otra.
- Cuando dos clases están en un mismo paquete, es porque se usan juntas. Por tanto, todas las clases de un paquete se usan conjuntamente: si se usa una, se usan todas.

A la inversa, *¿cuándo se ponen en paquetes diferentes?*

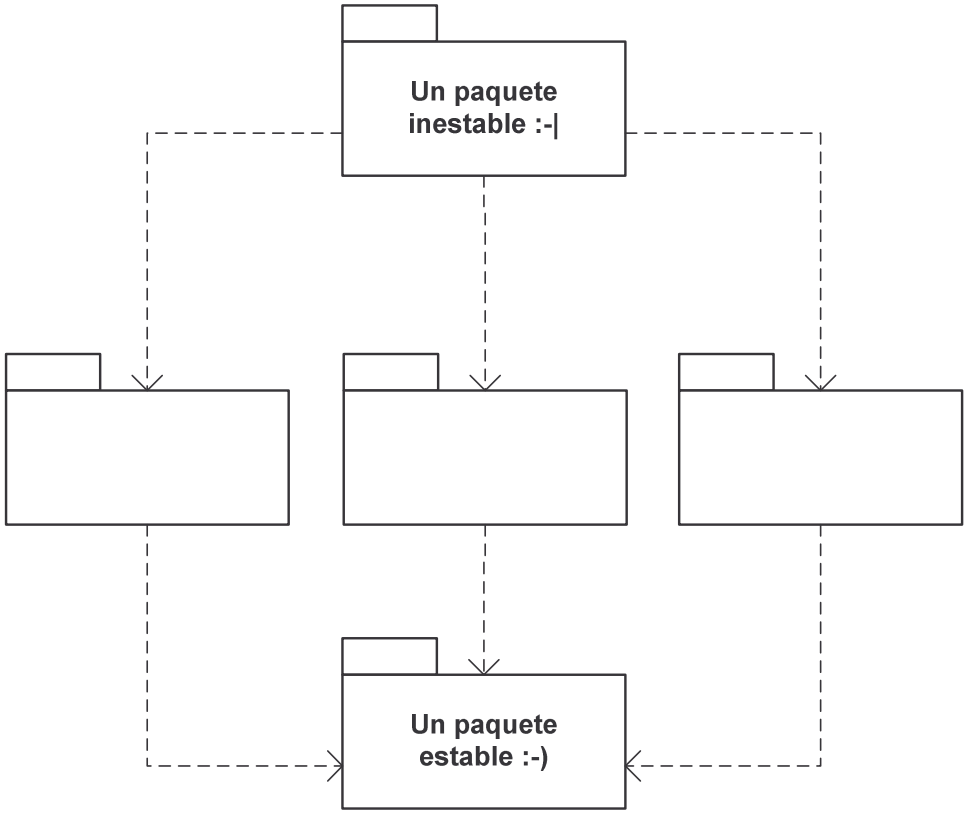
- Aquéllas clases que no siempre se usen conjuntamente con las demás clases del paquete son candidatas para abandonar el paquete (quizá para ir a un subpaquete más específico).

Las clases de un paquete se verán afectadas por los mismos tipos de cambios y las modificaciones necesarias para realizar un cambio concreto deberán estar localizadas en un único paquete.

Estabilidad:
El acoplamiento entre paquetes

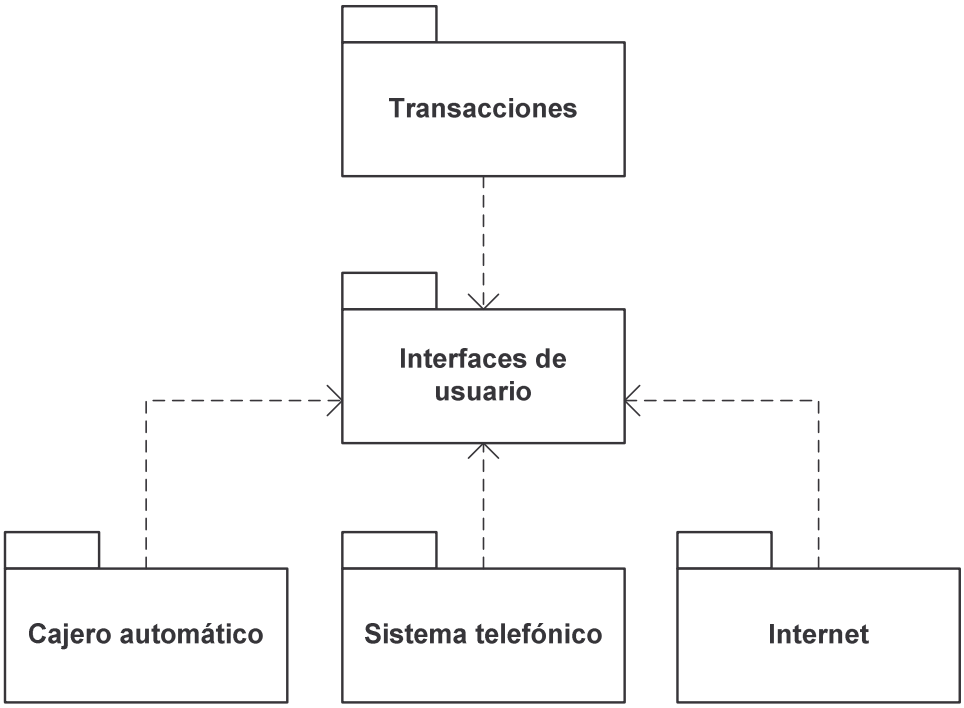
Para facilitar el desarrollo de un sistemas complejo, resulta aconsejable que las distintas partes del sistema sean lo más independientes posible:

Cuanto menos dependa un paquete de otros paquetes, mejor.



- Un paquete es más estable cuando depende de menos paquetes.
- Cuando queramos que un paquete sea flexible (esto es, fácil de cambiar), mejor si hay pocos paquetes que dependan de él.
- Un paquete debe ser más estable cuanto más abstracto sea (un paquete estable y concreto se vuelve rígido).

Si dibujamos un diagrama con las dependencias existentes entre los paquetes, el diagrama **no debe tener ciclos**:

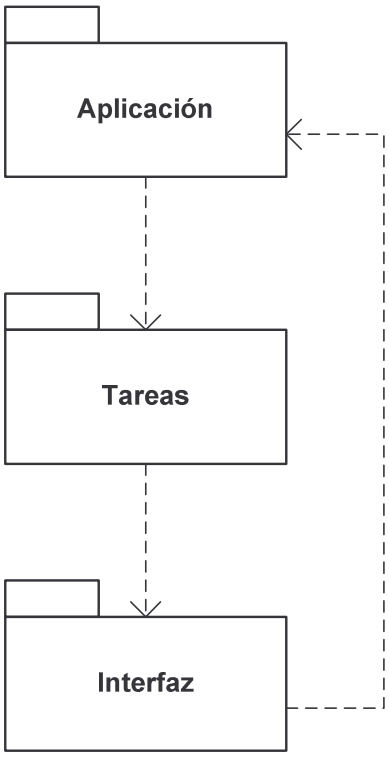


¿Qué efectos ocasiona un ciclo?

Cuando trabajamos
en la construcción de la interfaz,
hemos de disponer de ciertos servicios
proporcionados por la aplicación.

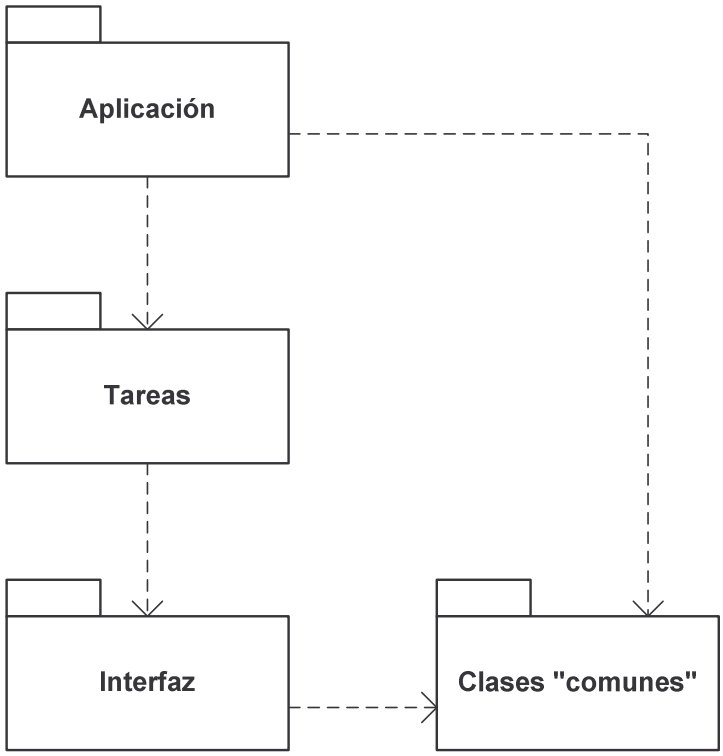
Esto hace que la interfaz dependa de todos los demás paquetes de la aplicación

Para probar el funcionamiento de la interfaz necesitamos disponer de una implementación de todos los demás paquetes y las pruebas dependerán del estado actual de esos paquetes (por lo que difícilmente se pueden considerar *pruebas de unidad*).



¿Cómo se rompe un ciclo entre dos paquetes?

1. Creando un nuevo paquete del que ambos dependan:



2. Aplicando el *principio de inversión de dependencias*: