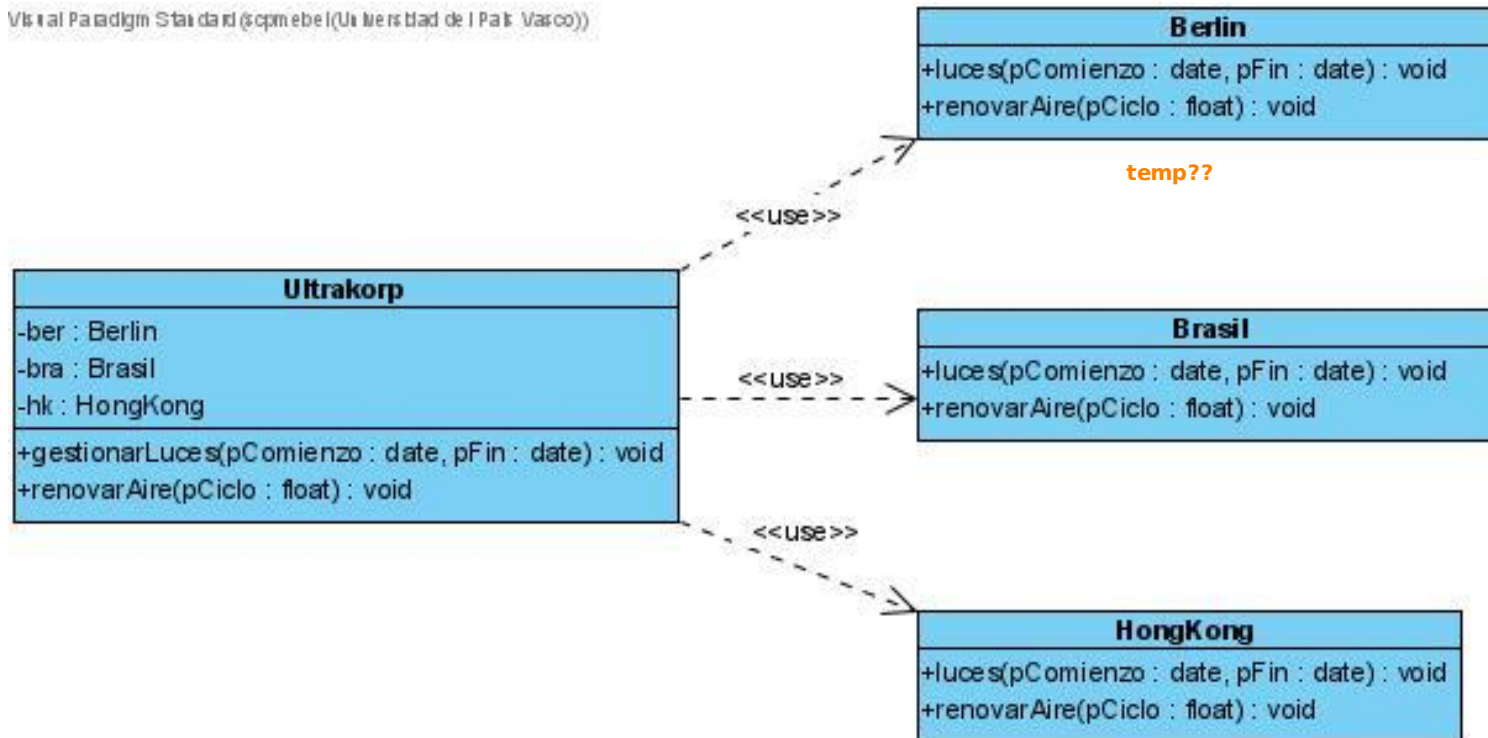


DIAGRAMA DE CLASES (SOLUCIÓN FACADE) (HAY DOS SOLUCIONES)

Bueno...

Visual Paradigm Standard (scpmebel@Universidad de I País Vasco))

Client...



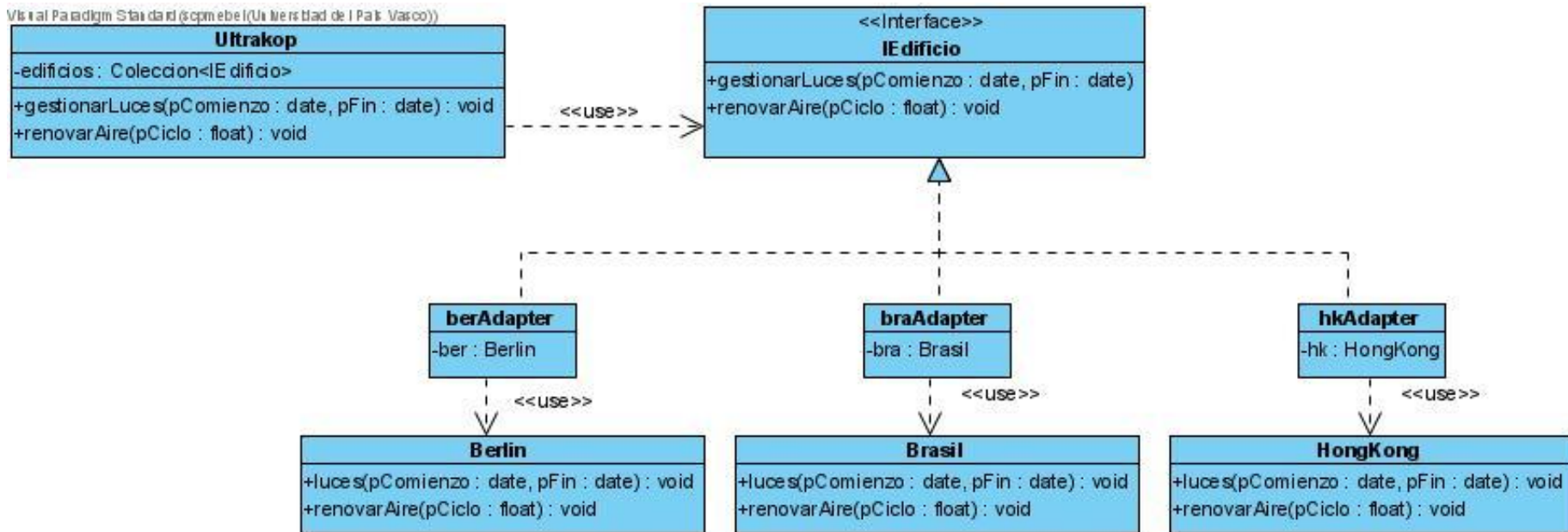
CÓDIGO (SOLUCIÓN FACADE)

```
public class Ultrakorp {  
  
    private Berlin ber;  
    private Brasil bra;  
    private HongKong hk;  
  
    public void gestionarLuces(Date pComienzo, Date  
pFin) {  
        ber.luces(pComienzo, pFin);  
        bra.luces(pComienzo, pFin);  
        hk.luces(pComienzo, pFin);  
    }  
  
    public void renovarAire(float pCiclo) {  
        ber.renovarAire(pCiclo);  
        bra.renovarAire(pCiclo);  
        hk.renovarAire(pCiclo);  
    }  
}
```

```
public class Berlin {  
    public void luces(Date pComienzo, Date pFin) {  
        System.out.println("Luces Berlin");  
    }  
    public void renovarAire(float pCiclo) {  
        System.out.println("Renovar aire Berlin");  
    }  
}  
  
public class Brasil {  
    public void luces(Date pComienzo, Date pFin) {  
        System.out.println("Luces Brasil");  
    }  
    public void renovarAire(float pCiclo) {  
        System.out.println("Renovar aire Brasil");  
    }  
}  
  
public class HongKong {  
    public void luces(Date pComienzo, Date pFin) {  
        System.out.println("Luces HongKong");  
    }  
    public void renovarAire(float pCiclo) {  
        System.out.println("Renovar aire  
HongKong");  
    }  
}
```

DIAGRAMA DE CLASES (SOLUCIÓN ADAPTER)

Visual Paradigm Standard (scpmeebe@Universidad de País Vasco)



CÓDIGO (SOLUCIÓN ADAPTER)

```
public class Ultrakop {
    private List<IEdificio> edificios = new
ArrayList<IEdificio>();
    public void gestionarLuces(Date pComienzo, Date pFin) {
        Iterator<IEdificio> it = edificios.iterator();
        while(it.hasNext())
        {
            it.next().gestionarLuces(pComienzo, pFin);
        }
    }
    public void renovarAire(float pCiclo) {
        Iterator<IEdificio> it = edificios.iterator();
        while(it.hasNext())
        {
            it.next().renovarAire(pCiclo);
        }
    }
}

public interface IEdificio {
    void gestionarLuces(Date pComienzo, Date pFin);
    void renovarAire(float pCiclo);
}
```

```
public class berAdapter implements IEdificio {
    private Berlin ber;
    @Override
    public void gestionarLuces(Date pComienzo, Date pFin) {
        ber.luces(pComienzo, pFin);
    }
    @Override
    public void renovarAire(float pCiclo) {
        ber.renovarAire(pCiclo);
    }
}

public class braAdapter implements IEdificio {
    private Brasil bra;
    @Override
    public void gestionarLuces(Date pComienzo, Date pFin) {
        bra.luces(pComienzo, pFin);
    }
    @Override
    public void renovarAire(float pCiclo) {
        bra.renovarAire(pCiclo);
    }
}

public class hkAdapter implements IEdificio {
    private HongKong hk;
    @Override
    public void gestionarLuces(Date pComienzo, Date pFin) {
        hk.luces(pComienzo, pFin);
    }
    @Override
    public void renovarAire(float pCiclo) {
        hk.renovarAire(pCiclo);
    }
}
```