Enunciado

Se debe diseñar un sistema para una empresa de telefonía.

En el contexto de esta empresa, existen clientes que poseen líneas telefónicas, las cuales se caracterizan por tener un número de teléfono y un plan asociado. Todas las líneas deben tener un número de teléfono único.

Cada plan determina el costo del minuto de llamada. La empresa factura por minuto y no distingue franjas horarias. El plan nunca cambia.

Desde una línea se pueden realizar varias llamadas a otra línea. A la empresa le interesa registrar la fecha, la hora, la duración y el costo de la llamada (el cual depende de la duración y el plan del cliente). Obviamente, no es posible llamar desde una línea a la propia línea y no es posible hablar con una línea, mientras se está hablando con otra.

Existen planes especiales, que además del precio, poseen una cantidad acotada de minutos disponibles para hablar (por ejemplo, sólo se pueden hablar 20 minutos por mes, a un precio de 29 centavos el minuto). Aquellos clientes que posean esta clase de plan, no podrán hablar más minutos que los que su plan indique.

A partir la descripción anterior, utilice la notación de diagramas de clases para elaborar el modelo conceptual, utilizando OCL para expresar todo lo que considere necesario.

Resolución

¿Cuál es el problema a modelar?

Tenemos una empresa de telefonía, que tiene clientes, los clientes realizarán llamadas para las cuales la empresa quiere facturar el monto correspondiente dependiendo del plan que tenga el cliente que efectuó la llamada. Además la empresa tiene planes especiales para los cuales no se puede hablar más de una cantidad determinada de minuto por mes...

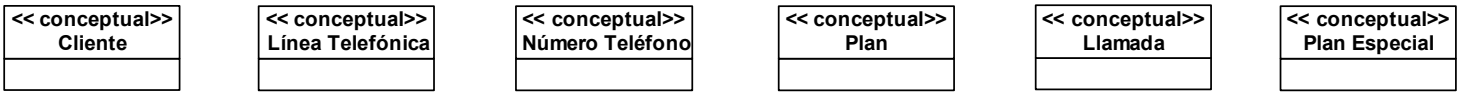
¿Cuáles son las entidades del dominio (o, conceptos) que se desprenden del enunciado?

En el contexto de esta empresa, existen clientes que poseen líneas telefónicas, las cuales se caracterizan por tener un número de teléfono y un plan asociado. Todas las líneas deben tener un número de teléfono único.

Cada plan determina el costo del minuto de llamada. La empresa factura por minuto y no distingue franjas horarias. El plan nunca cambia.

Desde una línea se pueden realizar varias llamadas a otra línea. A la empresa le interesa registrar la fecha, la hora, la duración y el costo de la llamada (el cual depende de la duración y el plan del cliente). Obviamente, no es posible llamar desde una línea a la propia línea y no es posible hablar con una línea, mientras se está hablando con otra.

Existen planes especiales, que además del precio, poseen una cantidad acotada de minutos disponibles para hablar (por ejemplo, sólo se pueden hablar 20 minutos por mes, a un precio de 29 centavos el minuto). Aquellos clientes que posean esta clase de plan, no podrán hablar más minutos que los que su plan indique.



Veamos las entidades que identificamos y pensemos qué atributos debe tener cada una de ellas... Nuevamente, volvamos al Enunciado...

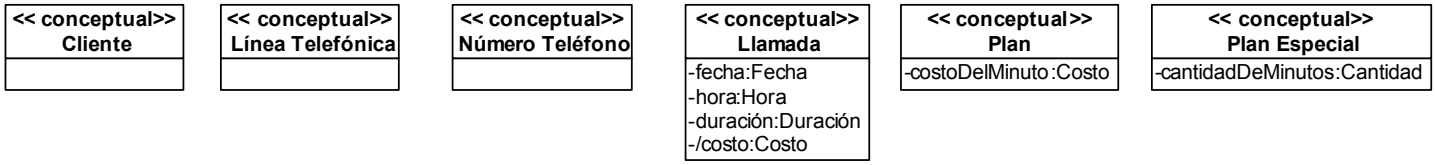
En el contexto de esta empresa, existen clientes que poseen líneas telefónicas, las cuales se caracterizan por tener un número de teléfono y un plan asociado. Todas las líneas deben tener un número de teléfono único.

Cada plan determina el costo del minuto de llamada. La empresa factura por minuto y no distingue franjas horarias. El plan nunca cambia.

Desde una línea se puede realizar varias llamadas a otra línea. A la empresa le interesa registrar la fecha, la hora, la duración y el costo de la llamada (el cual depende de la duración y el plan del cliente). Obviamente, no es posible llamar

desde una línea a la propia línea y no es posible hablar con una línea, mientras se está hablando con otra.

Existen planes especiales, que además del precio, poseen una cantidad acotada de minutos disponibles para hablar (por ejemplo, sólo se pueden hablar 20 minutos por mes, a un precio de 29 centavos el minuto). Aquellos clientes que posean esta clase de plan, no podrán hablar más minutos que los que su plan indique.



Una vez identificadas las Clases y sus atributos, veamos cómo se relacionan entre ellas...

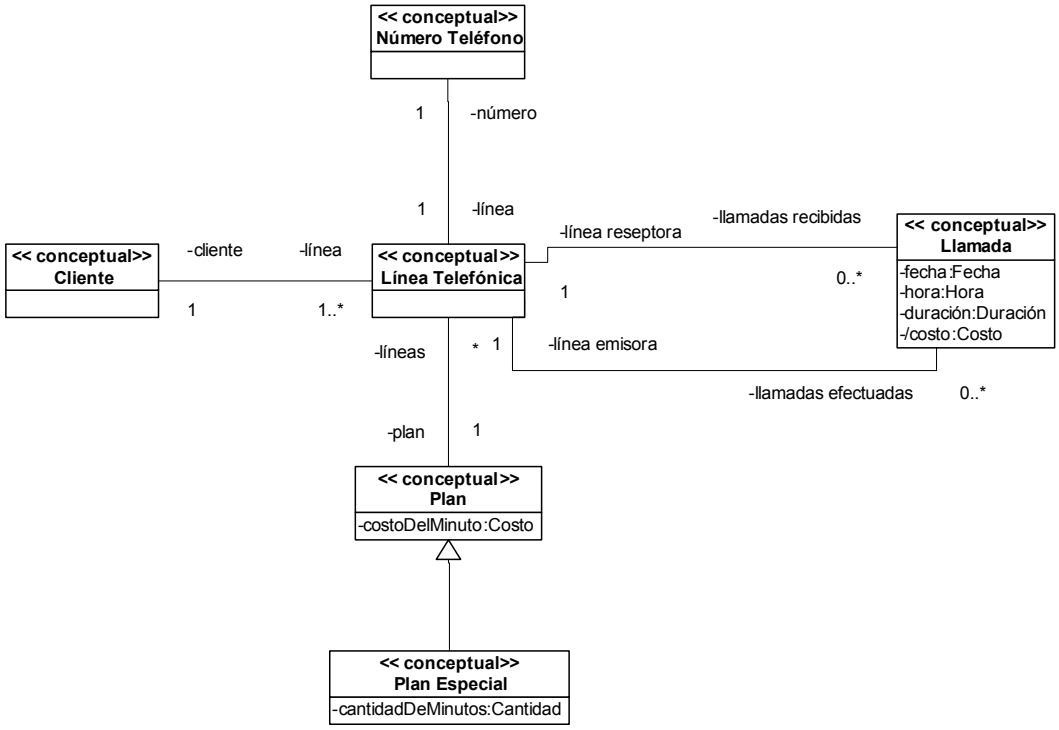
En el contexto de esta empresa, existen clientes que poseen líneas telefónicas, las cuales se caracterizan por tener un número de teléfono y un plan asociado. Todas las líneas deben tener un número de teléfono único.

Cada plan determina el costo del minuto de llamada. La empresa factura por minuto y no distingue franjas horarias. El plan nunca cambia.

Desde una línea se puede realizar varias llamadas a otra línea. A la empresa le interesa registrar la fecha, la hora, la duración y el costo de la llamada (el cual depende de la duración y el plan del cliente). Obviamente, no es posible llamar desde una línea a la propia línea y no es posible hablar con una línea, mientras se está hablando con otra.

Existen planes especiales, que además del precio, poseen una cantidad acotada de minutos disponibles para hablar (por ejemplo, sólo se pueden hablar 20 minutos por mes, a un precio de 29 centavos el minuto). Aquellos clientes que posean esta clase de plan, no podrán hablar más minutos que los que su plan indique.

De esta manera, tenemos el siguiente diagrama de Clases Conceptuales:



Comentarios sobre el modelo propuesto:

Variantes Válidas:

- El número de teléfono podría modelarse como atributo de la clase conceptual Línea Telefónica y dejaríamos tener la clase conceptual "Número Teléfono".

Asumimos que:

- La fecha, hora y duración de una llamada están en minutos y son sumables entre sí.
- El tipo Fecha tiene atributos mes y año. Por ejemplo, si f es una fecha podremos saber su año y mes vía f.año y f.mes respectivamente.
- La cantidad de minutos de un plan especial es por mes.

Aclaración:

- El carácter "-" delante de los atributos de las clases conceptuales en el diagrama, no tiene ningún significado en el Modelo Conceptual. Por el contrario, cuando un atributo se encuentre precedido por el carácter "/", nos estaremos refiriendo simpre a atributos derivados.

Volviendo a ver el enunciado ¿qué estaría faltando modelar que no se pueda deducir del modelo de clases conceptuales?

En el contexto de esta empresa, existen clientes que poseen líneas telefónicas, las cuales se caracterizan por tener un número de teléfono y un plan asociado. Todas las líneas deben tener un número de teléfono único.

Cada plan determina el costo del minuto de llamada. La empresa factura por minuto y no distingue franjas horarias. El plan nunca cambia.

Desde una línea se puede realizar varias llamadas a otra línea. A la empresa le interesa registrar la fecha, la hora, la duración y el costo de la llamada (el cual depende de la duración y el plan del cliente). Obviamente, no es posible llamar desde una línea a la propia línea y no es posible hablar con una línea, mientras se está hablando con otra.

Existen planes especiales, que además del precio, poseen una cantidad acotada de minutos disponibles para hablar (por ejemplo, sólo se pueden hablar 20 minutos por mes, a un precio de 29 centavos el minuto). Aquellos clientes que posean esta clase de plan, no podrán hablar más minutos que los que su plan indique.

Usamos OCL expresarlo…

1) "El costo de la llamada (el cual depende de la duración y el plan del cliente)."

En el contexto de Llamada:

```
context Llamada::costo: Costo
derive: self.costo = self.duración * self.línea llamadora.plan.costoDelMinuto
```

Otra manera de expresarlo podría ser en el contexto de Línea Telefónica:

```
context Línea Telefónica
inv: self.llamadas efectuadas->forAll ( l1 | l1.costo = l1.duración * self.plan.costoDelMinuto)
```

2) "No es posible llamar desde una línea a la propia línea."

En el contexto de Línea Telefónica:

```
context Línea Telefónica
inv: self.llamadas efectuadas -> forAll ( l1 | l1.línea receptora <> self)
```

En el contexto de Llamada:

```
context Llamada
inv: self.línea llamadora <> self.línea línea receptora
```

3) "No es posible hablar con una línea, mientras se está hablando con otra."

```
context Llamada
inv: Llamada.allInstances() -> forAll ( l1, l2 | l1 <> l2 and
                                    ( l1.línea llamadora = l2.línea llamadora
                                    or l1.línea llamadora = l2.línea receptora
                                    or l1.línea receptora = l2.línea receptora )
                               implies
                                    l1.fecha + l1.hora + l1.duración < l2.fecha + l2.hora
                                    or l2.fecha + l2.hora + l2.duración < l1.fecha + l1.hora )
```

o, dicho de otra forma:

```
not (
        (l1.fecha + l1.hora + l1.duración >= l2.fecha + l2.hora
        and l1.fecha + l1.hora + l1.duración <= l2.fecha + l2.hora + l2.duración)
 and
        (l2.fecha + l2.hora + l2.duración >= l1.fecha + l1.hora
        and l2.fecha + l2.hora + l2.duración <= l1.fecha + l1.hora + l1.duración)
)
```

4) "Aquellos clientes que posean Planes especiales, no podrán hablar más minutos que los que su plan indique."

```
context Línea Telefónica
inv: self.plan.oclIsTypeOf(Plan Especial) implies
  self.llamadas efectuadas -> forAll( l1 |
        self.llamadas efectuadas -> select ( l2 | l1.fecha.año = l2.fecha.año
        and l1.fecha.mes = l2.fecha.mes) -> collect (duración) - > sum() <=
        self.plan.oclAsType(Plan Especial).cantidadDeMinutos)
```

El antecedente del implies también podría ser escrito de la siguiente forma:

```
not ( self.plan.oclIsTypeOf(Plan Especial).oclIsUndefined() )
```

También puede expresarse de la siguiente manera:

```
context Línea Telefónica
inv: if self.plan.oclIsTypeOf(Plan Especial)
        then self.llamadas efectuadas -> forAll( l1 | self.llamadas efectuadas -> select ( l2 |
                l1.fecha.año = l2.fecha.año and
                l1.fecha.mes = l2.fecha.mes) -> collect (duración) - > sum() <=
                self.plan.oclAsType(Plan Especial).cantidadDeMinutos)
        else true
        endif
```

O bien, en el contexto de Plan Especial:

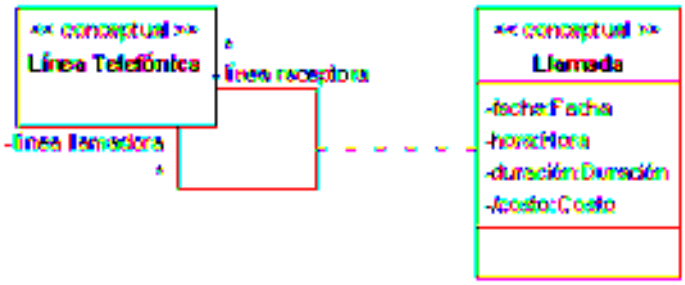
```
context Plan Especial
inv: self.líneas -> forAll( lt | lt.llamadas efectuadas -> forAll( l1 |
        self.llamadas efectuadas -> select ( l2 | l1.fecha.año = l2.fecha.año and
l1.fecha.mes = l2.fecha.mes).duración -> sum() <= self.cantidadDeMinutos)
```

En lugar de la forma corta del collect (".duracion"), podría usarse la forma completa:

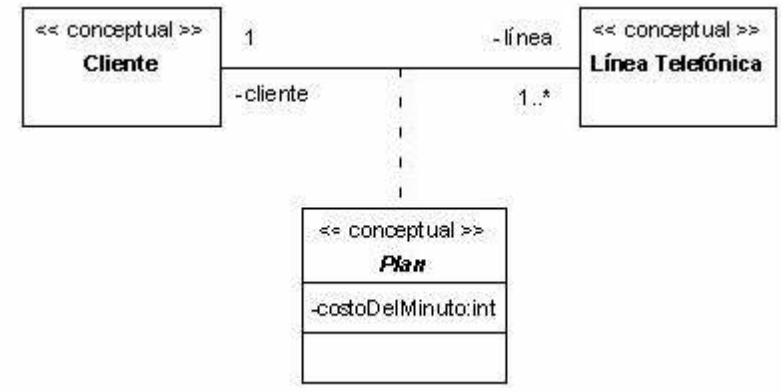
```
context Plan Especial
inv: self.líneas -> forAll( lt | lt.llamadas efectuadas -> forAll( l1 |
        self.llamadas efectuadas -> select ( l2 | l1.fecha.año = l2.fecha.año and
l1.fecha.mes = l2.fecha.mes) -> collect (duración) -> sum() <= self.cantidadDeMinutos)
```

Errores comunes

- Clase conceptual *Llamada* como clase de asociación entre *Línea Telefónica* y *Línea Telefónica*. Se debe poder registrar más de una llamada entre dos líneas telefónicas distintas.



- Clase conceptual *Plan* como clase de asociación entre *Cliente* y *Línea Telefónica*. La existencia de los planes en la empresa de telefonía no depende de la existencia de líneas telefónicas para clientes.



- Modelar la clase conceptual Empresa de Telefonía.

- El atributo costo en la clase conceptual Llamada no derivado.

- Falta de roles.

- Falta de multiplicidades.

- Falta de estereotipo: <<clase conceptual>>, <<c.c.>> o <<conceptual>>.

- Agregar atributos a clases conceptuales que no se desprendan del enunciado, por ejemplo, nombre y DNI a Cliente.

- Ausencia de suposiciones que se dan por sentado en la resolución.

- Nombres no apropiados o poco declarativos, por ejemplo, Línea en lugar de Línea Telefónica, Número en lugar de Número de Teléfono, etc.