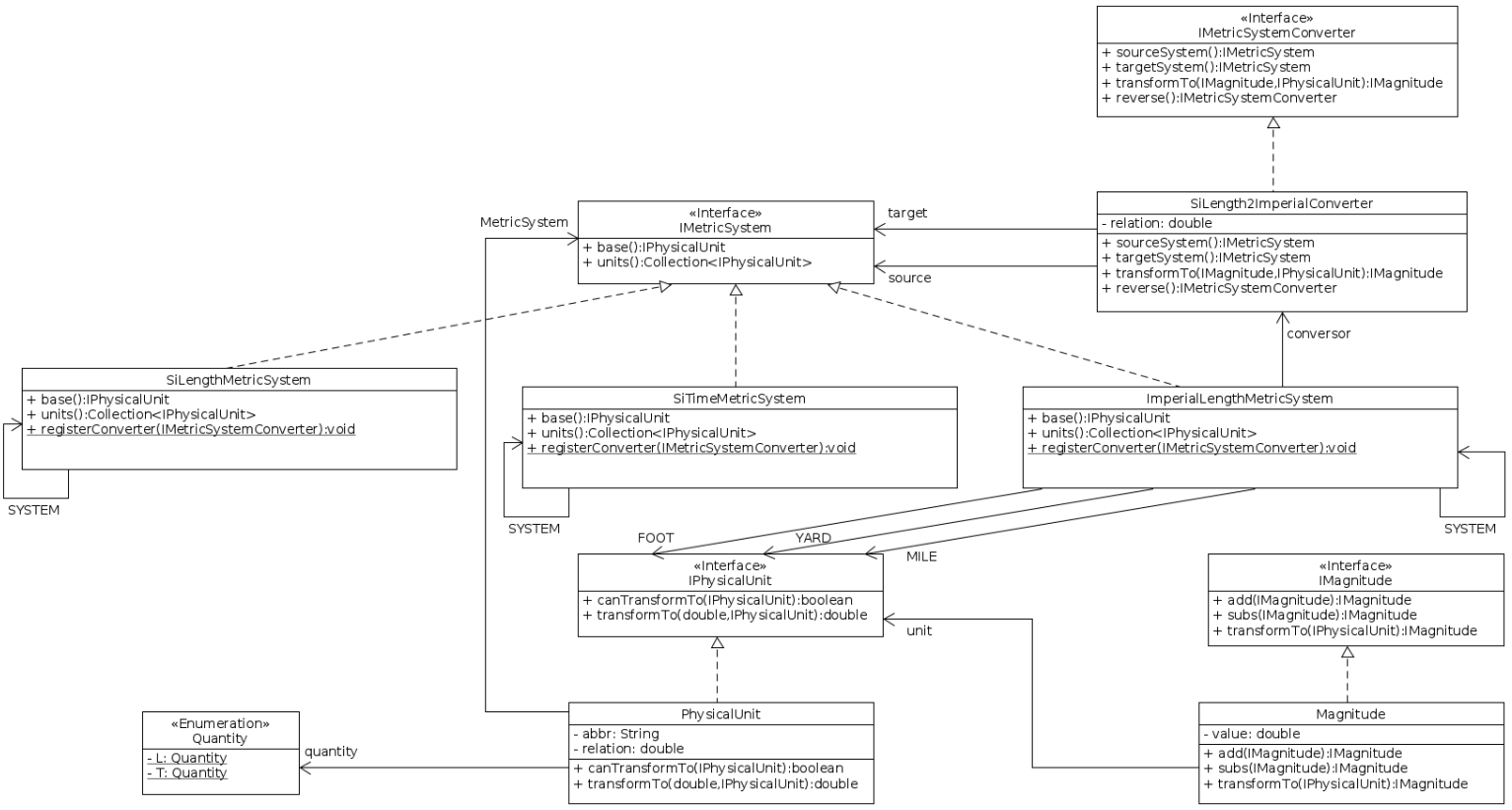

Este es nuestro diagrama de clases simplificado. Además de omitirse getters y setters, se han omitido asociaciones como las que se dan entre los 3 tipos de sistemas y la clase Quantity y se ha ejemplificado con ImperialLengthMetricSystem las asociaciones de los otros dos sistemas con IphysicalUnit y con el conversor SiLength2ImperialConverter. No se indica la cardinalidad de las asociaciones pues son todas 0..1

En lo que respecta a la extensibilidad del diseño, para añadir nuevas unidades a un sistema existente, como por ejemplo añadir la unidad *chain* al ImperialLengthMetricSystem, bastaría con crear una nueva PhysicalUnit que fuera *chain* y añadirla al ImperialLengthMetricSystem, esto es, establecer una asociación como la asociación FOOT.

Para añadir la cantidad Masa al Sistema Métrico Internacional, habría que aumentar la enum de Quantity añadiendo Masa(M). En el caso de querer implementar completamente el sistema métrico, hay que crear una nueva clase que implemente ImetricSystem que sea SiMassMetricSystem por ejemplo y establecer las asociaciones correspondientes, similares a las existentes en los otros sistemas métricos.

Finalmente, en lo referente a las desventajas o limitaciones del diseño, cabe destacar que para añadir nuevas conversiones, habría que crear nuevas clases que implementaran esta conversión, cuando podría haberse diseñado de manera que fuera más sencillo establecer esas conversiones.

En lo que respecta a los apartados 1-4, estas son las salidas correspondientes de cada uno de ellos (en orden ascendente)

```
m L
true
false
1000 m en km: 1.0
Quantities t and L are not compatible
```

```
[mm L, m L, km L]
Base = m L
false
```

```
12512.5 m L
12.5 km L
12500.0 m L
Quantities t and L are not compatible
```

```
Cannot transform km L to mi L
En millas = 6.213711922348486 mi L
En m = 10000.000000000002 m L
```