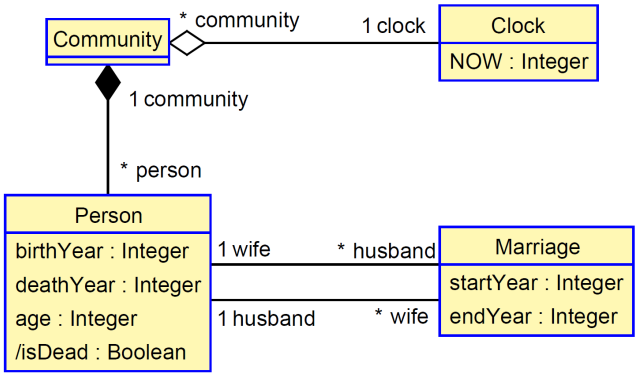
Q5. Ejercicio del tema 3.

Supongamos el siguiente diagrama UML que representa la estructura de una comunidad compuesta por personas, que a lo largo de su vida pueden casarse, divorciarse o enviudar. En cuanto al comportamiento del sistema, las personas pueden atravesar varios estados a lo largo de su vida: soltero, casado, divorciado o viudo. Igualmente, conforme van cumpliendo años van atravesando diversas etapas (o estados): niño, joven, adulto, anciano y, por último, fallecido cuando mueren.



(A) En primer lugar, se pide modelar correctamente la estructura de dicho sistema, incluyendo aquellos elementos o restricciones que se consideren necesarias para representar lo que sucede en realidad, incluyendo al menos las siguientes:

Restricciones aléticas ("alethic constraints" o "alethic invariants"):

- Una persona no puede fallecer antes de nacer.
- Un matrimonio no puede acabar antes de empezar.
- Los muertos no pueden estar casados.
- No se puede nacer en el futuro: Clock.NOW tiene que ser mayor o igual que cualquier fecha de nacimiento.
- No puede haber matrimonios futuros ("acordados"): Clock.NOW tiene que ser siempre mayor o igual que cualquier fecha de comienzo de un matrimonio.
- No se puede divorciar una persona de alguien con quien no esté actualmente casado.

Restricciones "deónticas" ("business rules" o "deontic constraints")

- No puede haber más que un reloj en toda la aplicación (y por tanto compartido por todas las comunidades)
- Una persona no puede estar casada consigo misma.
- Monogamia: Una persona no puede tener más de un matrimonio activo en un momento dado.
- Los niños no pueden estar casados.
- No se permite ni la eutanasia ni el suicidio.
- La edad en las que las personas cambian de niño a joven, de joven a adulto o de adulto a anciano puede depender de la comunidad a la que pertenezca la persona.

(Se valorará incluir restricciones adicionales que no puedan derivarse de las anteriores)

(B) Especifica en OCL las siguientes operaciones de consulta (*queries*) sobre la clase Person:

- isMarried():Boolean que devuelve si una persona está actualmente casada o no
- marriages():Integer que devuelve el número de matrimonios que ha tenido una persona a lo largo de su vida (incluido el actual, si es que estás casado).

(C) Modela el comportamiento de dicho sistema en USE. Para ello se pide:

- Definir las operaciones necesarias para modelar el paso del tiempo, y que las personas puedan, casarse, divorciarse, cumplir años y fallecer.
- Para que una persona pueda casarse con otra, es preciso pedirle su consentimiento y que acceda. Igualmente, para que una persona pueda divorciase de otra, es preciso que acceda.
- Especificar en OCL, para cada una de dichas operaciones, sus pre- y post-condiciones.
- Especificar en SOIL, para cada una de dichas operaciones, su comportamiento.
- Especificar, para cada uno de los tipos definidos en el modelo, las máquinas de estado que se consideren oportunas.
- Crear un modelo con al menos 10 personas, y especificar al menos dos posibles ejecuciones diferentes en donde se creen al menos 4 matrimonios distintos y fallezcan al menos 3 personas. Generar los correspondientes diagramas de secuencia y de comunicación en USE.

(D) Responde a las siguientes cuestiones:

- ¿Sería posible que se casasen personas de diferentes comunidades? En ese caso, ¿habría algún problema en la forma en la que habéis especificado las restricciones y las máquinas de estado de las personas, para que, a pesar de estar casadas, siempre se respeten las reglas de las dos comunidades?
- De acuerdo a tu modelo, ¿cuál sería el estado civil de una persona muerta? ¿Qué ocurre en la realidad? Es decir, de acuerdo a nuestro sistema legal en España, ¿Cuál es el estado civil de una persona fallecida? ¿Es lo mismo en otros países?
- ¿Has comprobado cuáles son los estados civiles que definen nuestras leyes? ¿Es lo mismo en todos los países? Si no, ¿cómo se debería modelar la aplicación que cada persona tuviera diferentes estados dependiendo de la comunidad a la cual perteneciera?
- En algunos países, casarse con menores no es delito civil, solo un problema ético. ¿Afectaría este hecho de alguna forma a tu modelo?

NOTAS:

— *El ejercicio se entregará en dos archivos: uno comprimido (zip) que contendrá los proyectos Visual Paradigm, Papyrus y USE creados al efecto, y otro en formato PDF con una memoria que incluya *todos* los diagramas de clases y objetos desarrollados, las restricciones definidas, así como aquellas explicaciones textuales que puedan considerarse necesarias por no estar claras en los diagramas o que sean necesarias para facilitar la comprensión del modelo, sus entidades y relaciones. Asimismo, contendrá las respuestas a las preguntas solicitadas en el apartado D.*

— *El nombre del fichero comprimido ha de contener el identificativo del grupo (con tres caracteres) y la cadena "Q4" separados por guiones bajos ("_"). Por ejemplo "B01_Q5.zip"*

— *Es importante que el fichero PDF sea autocontenido, y tenga toda la información para realizar la evaluación (códigos USE y SOIL, diagramas, etc.). El nombre del fichero PDF debe obedecer a las mismas reglas que las definidas para el fichero ZIP (p.ej., "B01_Q5.pdf").*

La evaluación del ejercicio se hará teniendo en cuenta los siguientes aspectos:

- Dominio de la notación gráfica y textual de UML para el modelo dinámico.
- Corrección del modelo desarrollado y de sus vistas (en caso de haberlas)
- Simplicidad del modelo
- Completitud y expresividad del modelo
- Fidelidad de la representación del dominio del problema descrito en el enunciado.