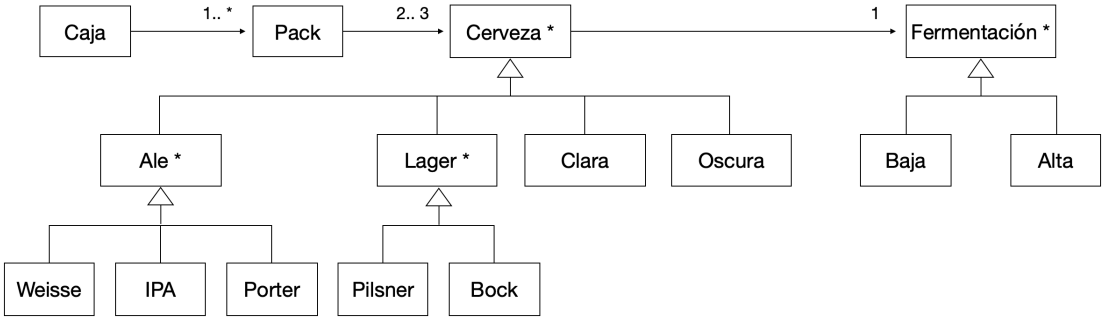
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

Métodos Formales para Ingeniería de Software

Examen Recuperatorio

24 de Noviembre de 2023

El siguiente diagrama modela información acerca de las características de las cervezas que elabora una empresa artesanal, así como también información sobre las formas de presentación en las que se venden (packs/cajas):



El tipo de toda cerveza es Ale o Lager y cada tipo se especializa en dos o más variedades. Asimismo, toda cerveza tiene una tonalidad, la cual es clara, oscura o intermedia (mezcla de clara y oscura). Por último, existen limitaciones con respecto a la cantidad de cervezas que integran los packs y cajas. Más específicamente, la venta de cervezas se realiza en packs de 2 o 3 cervezas, o en cajas de packs que contabilizan un total de entre 2 y 6 cervezas.
Considere el siguiente modelo en Alloy, correspondiente al dominio antes presentado. Asuma que el mismo es correcto y fue validado en un contexto de modelado estático:

```
abstract sig Cerveza { temperatura: one Fermentacion }

abstract sig Ale, Lager extends Cerveza {}

sig Clara, Oscura in Cerveza {}

sig Weisse, IPA, Porter extends Ale {}

sig Pilsner, Bock extends Lager {}

abstract sig Fermentacion {}

one sig Alta, Baja extends Fermentacion {}

sig Caja { packs: some Pack }

sig Pack { cervezas: some Cerveza }

fact {all c: Cerveza | (c in (Clara-Oscura)) or (c in (Oscura-Clara)) or (c in (Clara & Oscura))}

fact {all p: Pack | (#p.cervezas = 2) or (#p.cervezas = 3)}

fact {all c: Caja | (#c.packs.cervezas >= 2) and (#c.packs.cervezas =< 6)}

// Los packs de una misma caja no comparten cervezas
fact {all c: Caja | no disj p1, p2: c.packs | some (p1.cervezas & p2.cervezas)}
```

a) Extienda el modelo brindado incorporando las siguientes restricciones:

- Las cervezas *Ale* se fermentan a temperatura alta, mientras que las *Lager* se fermentan a temperatura baja.
- Las cervezas dentro del mismo pack son de la misma variedad o poseen la misma tonalidad.
- Las cervezas *Weisse* y *Pilsner* son claras, mientras que las *IPA* y *Bock* son intermedias, y las *Porter* son oscuras.

Brinde comandos para validar el modelo resultante de añadir dichas restricciones. Para cada comando, deberá explicarse el *propósito* del mismo, el *resultado esperado* y el *resultado obtenido* al ejecutarlo. En particular, para aquellos comandos que no generen instancias, deberán explicarse los motivos por los cuales no lo hacen.

b) Realice una copia del archivo *.als* elaborado hasta el momento, y trabaje sobre el nuevo archivo. Incorpore al modelo resultante del inciso anterior el siguiente predicado, el cual modela el comportamiento de *cambiar una cerveza de un pack por otra cerveza que esté fuera del pack*. Esta acción es posible siempre y cuando el pack no esté en una caja.

```
pred cambiarCerveza[c1, c2: Cerveza, p1, p2: Pack]{
    (some (c2 & p1.cervezas)) and
    (c2 in p2.cervezas)
}
```

Utilice el analizador para validar si la definición del predicado `cambiarCerveza` es correcta, considerando el modelo resultante del inciso *a*) y la descripción brindada para el predicado.
IMPORTANTE: En este inciso no deberán realizarse modificaciones sobre el archivo resultante del inciso *a*) más allá del añadido del predicado `cambiarCerveza` y los comandos utilizados para validarlo.

Deberá dejarse registro de cada comando utilizado en el proceso de verificación, teniendo en cuenta las siguientes consideraciones:

- Para cada comando, deberá indicarse el *propósito*, el *resultado esperado* de su ejecución (indicando si se espera que el predicado tenga éxito o no, a partir de la descripción brindada para el mismo) y el *resultado obtenido* al ejecutarlo.
- Para aquellos comandos que no generen instancias, deberán explicarse los motivos por los cuales no lo hacen.
- Para aquellos comandos que generen instancias en las que se observan irregularidades, deberá dejarse registro de dicha instancia (por ejemplo, mediante una captura de pantalla), describiendo cuáles son las irregularidades o problemas allí observados e indicando cuál fue el comando utilizado para generar dicha instancia; en caso de ser necesario, deberá indicarse también el número de instancia generada por dicho comando (por ejemplo, si se trata de la primera instancia, de la segunda instancia, etc.).

c) Realice una copia del archivo *.als* elaborado hasta el momento, y trabaje sobre el nuevo archivo. Realice los cambios necesarios en el predicado `cambiarCerveza` y/o en el modelo de manera tal que la nueva definición del predicado respete la descripción brindada anteriormente.
IMPORTANTE: La versión modificada del predicado deberá tener el siguiente encabezado:

`cambiarCervezaV2[c1, c2: Cerveza, p1, p2: Pack]`

Una vez efectuados los cambios necesarios, valide el predicado resultante considerando al menos 7 *casos de éxito* significativos y al menos 7 *casos de no éxito* significativos. En particular, para cada comando definido, deberá indicarse el *propósito* y el *resultado esperado* de su ejecución

(indicando si se espera que el predicado tenga éxito o no, a partir de la descripción brindada para el mismo) y el *resultado obtenido* al ejecutarlo.

d) Realice una copia del archivo *.als* elaborado hasta el momento, y trabaje sobre el nuevo archivo. Defina un predicado que modele el comportamiento de *añadir un pack a una caja* respetando el siguiente encabezado:

```
agregarPack[p: Pack, c1, c2: Caja]
```

Si el pack a añadir posee 2 cervezas, ninguna cerveza del pack a ser añadido debe tener la misma tonalidad que alguna cerveza de los packs restantes de la caja. Por otra parte, si el pack a añadir posee 3 cervezas, ninguna cerveza del pack a ser añadido debe ser de la misma variedad que alguna cerveza de los packs restantes de la caja.

Deberá explicitarse toda pre y post-condición asociada a la operación, incluyendo las indicadas en la descripción brindada, así como también las condiciones de marco y cualquier otra condición que sea necesaria.

Valide el predicado definido, considerando al menos 7 *casos de éxito* significativos y al menos 5 *casos de no éxito* significativos. En particular, para cada comando definido, deberá indicarse claramente el *propósito*, el *resultado esperado* de su ejecución (indicando si se espera que el predicado tenga éxito o no, a partir de la descripción brindada para el mismo) y el *resultado obtenido* al ejecutarlo.

Observaciones Generales:

- Puede brindarse cualquier otra especificación (hecho, aserción, predicado, función, etc.) adicional que considere necesaria y sea sensata.
- Como se mencionó anteriormente, deberá dejarse registro de todo comando utilizado para validar el modelo, las restricciones, predicados y/o funciones definidos.
- En caso de utilizar el evaluador para complementar la validación del modelo, deberá dejarse registro (mediante capturas de pantalla) de la verificación realizada con el evaluador, indicando el comando utilizado para generar la instancia sobre la cual se utilizó el mismo.