Tópicos II – Victor Martinez – TP Patrones

Requerimiento 1: Exportar las preguntas en JSon y XML.

Se aplica el patrón Visitor sobre las clases **Pregunta** y **Respuesta** para exportar los datos en los formatos requeridos.

Para ello se genera la clase abstracta **Visitor,** donde se declaran los métodos **visitPregunta(Preguntas)** y **visitRespuesta(Respuestas)**. Luego se crean las clases **VisitorJSon** y **VisitorXML** que heredan de Visitor y en donde se implementan los métodos visitPregunta y visitRespuesta, estos métodos reciben como parámetro el objeto que se desea exportar (Pregunta o Respuesta) y cada clase realiza la exportación en el formato correspondiente.

Métodos implementados en las clases VisitorJSon y VistorXML (faltan algunos atributos):

VisitorJSon

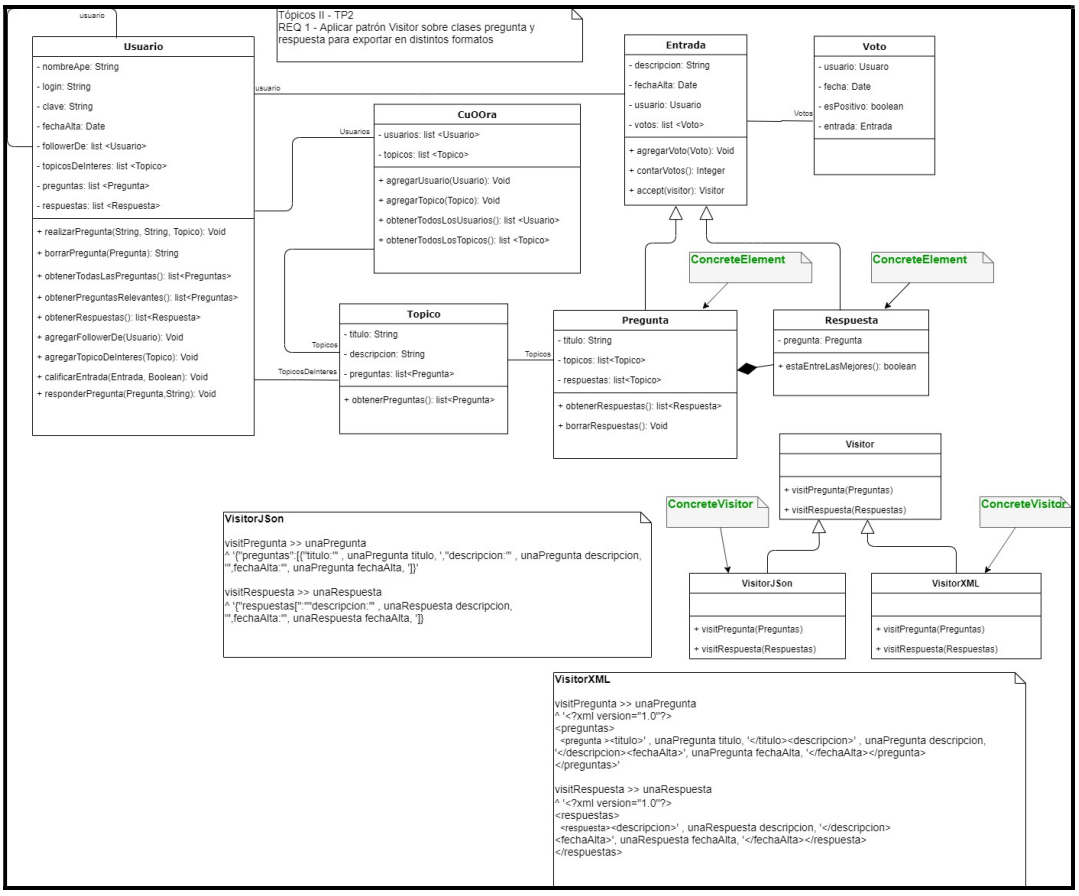
```
visitPregunta >> unaPregunta
^ '{"preguntas":[{"titulo:"' , unaPregunta titulo, ',"descripcion:"' , unaPregunta descripcion,
'",fechaAlta:"', unaPregunta fechaAlta, ']}'

visitRespuesta >> unaRespuesta
^ '{"respuestas[":""descripcion:"' , unaRespuesta descripcion,
'",fechaAlta:"', unaRespuesta fechaAlta, ']}
```

VisitorXML

```
visitPregunta >> unaPregunta
^ '<?xml version="1.0"?>
<preguntas>
  <pregunta ><titulo>' , unaPregunta titulo, '</titulo><descripcion>' , unaPregunta descripcion,
'</descripcion><fechaAlta>', unaPregunta fechaAlta, '</fechaAlta></pregunta>
</preguntas>'

visitRespuesta >> unaRespuesta
^ '<?xml version="1.0"?>
<respuestas>
  <respuesta><descripcion>' , unaRespuesta descripcion,
'</descripcion><fechaAlta>', unaRespuesta fechaAlta, '</fechaAlta></respuesta>
</respuestas>
```



Requerimiento 1

Requerimiento 2: Personalización del Home.

Se utiliza el patrón Strategy para aplicar las diferentes personalizaciones del home.

Para ello se crea la clase abstracta **Strategy** de define el método **obtenerPreguntasRelevantes**, luego se crean las clases **Social**, **Topicos**, y **Relevantes** que heredan de Strategy e implementan el método **obtenerPreguntasRelevantes** con el criterio correspondiente.

Métodos de implementados en las clases Social, Topicos y Relevantes:

Social

```
obtenerPreguntasRelevantes
|pregRelevantes |
        pregRelevantes := OrderedCollection new.
        followerDe do: [:a | pregRelevantes addAll: a obtenerTodasLasPreguntas   ].

        ^ pregRelevantes.
```

Topicos

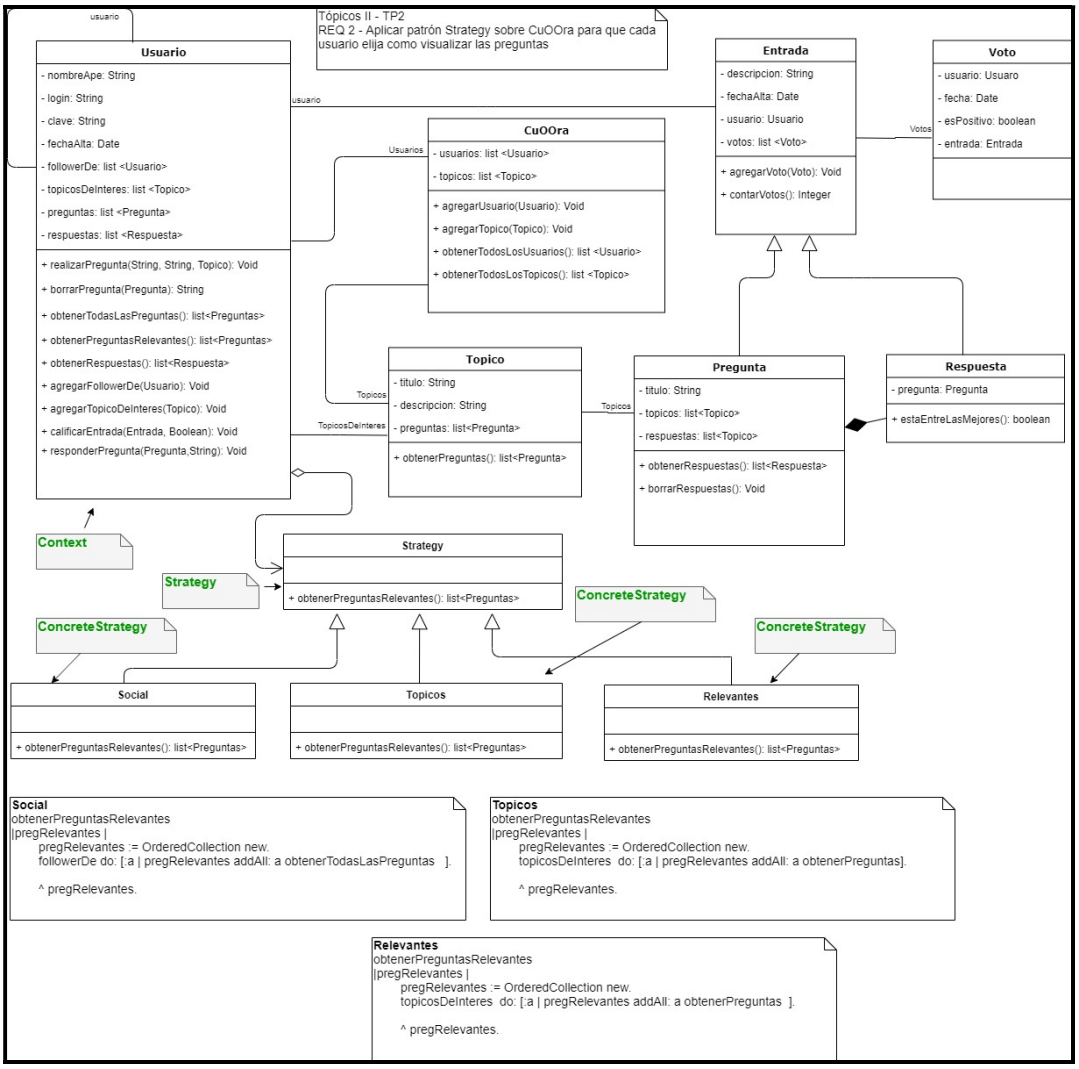
```
obtenerPreguntasRelevantes
|pregRelevantes |
        pregRelevantes := OrderedCollection new.
        topicosDeInteres  do: [:a | pregRelevantes addAll: a obtenerPreguntas].
```

^ pregRelevantes.

Relevantes

```
obtenerPreguntasRelevantes
|pregRelevantes |
        pregRelevantes := OrderedCollection new.
        topicosDeInteres  do: [:a | pregRelevantes addAll: a obtenerPreguntas  ].

        ^ pregRelevantes.
```



Requerimiento 2

Requerimiento 3: Estados en la pregunta.

Se utiliza el patrón State para aplicar diferentes estados a las preguntas.

Para ello se crea la clase abstracta **PreguntaState** que define los métodos **avanzarEstado** y **retrocederEstado**. Luego se crean las clases **Creada**, **Activa** y **Cerrada** que implementan estos métodos con el criterio requerido para los cambios de estado.

Además, se modifica el método **agregarRespuesta** de la clase **Pregunta** para validar el estado de la pregunta antes de generar una respuesta y también el **constructor** para instanciar el estado al momento de la creación.

Métodos implementados en las clases Creada, Activa y Cerrada:

Creada

```
avanzarEstado >> unaPregunta
unaPregunta estado: Activa new.
^ 'Pregunta en estado Activa'.

retrocederEstado >> unaPregunta
^self error:  'No se puede retroceder del estado Creada'.
```

Activa

```
avanzarEstado >> unaPregunta
unaPregunta estado: Cerrada new.
^ 'Pregunta en estado Cerrada'.

retrocederEstado >> unaPregunta
(unaPregunta estado isMemberOf: Activa)
                    ifFalse:   [ ^self error:  'Solo se puede volver a estado Creada desde estado
Activa'.  ].
(unaPregunta respuestas isEmpty)
                    ifFalse:   [ ^self error:  'Solo se puede volver a estado creada si la pregunta
no tiene respuestas'.  ].
unaPregunta estado: Creada new.
^ 'Pregunta en estado Creada'.
```

Cerrada

```
avanzarEstado >> unaPregunta
^self error:  'No se puede avanzar del estado Cerrada'.

retrocederEstado >> unaPregunta
^self error:  'No se puede retroceder del estado Cerrada'.
```

Modificación del constructor para agregar el estado de la pregunta

Pregunta

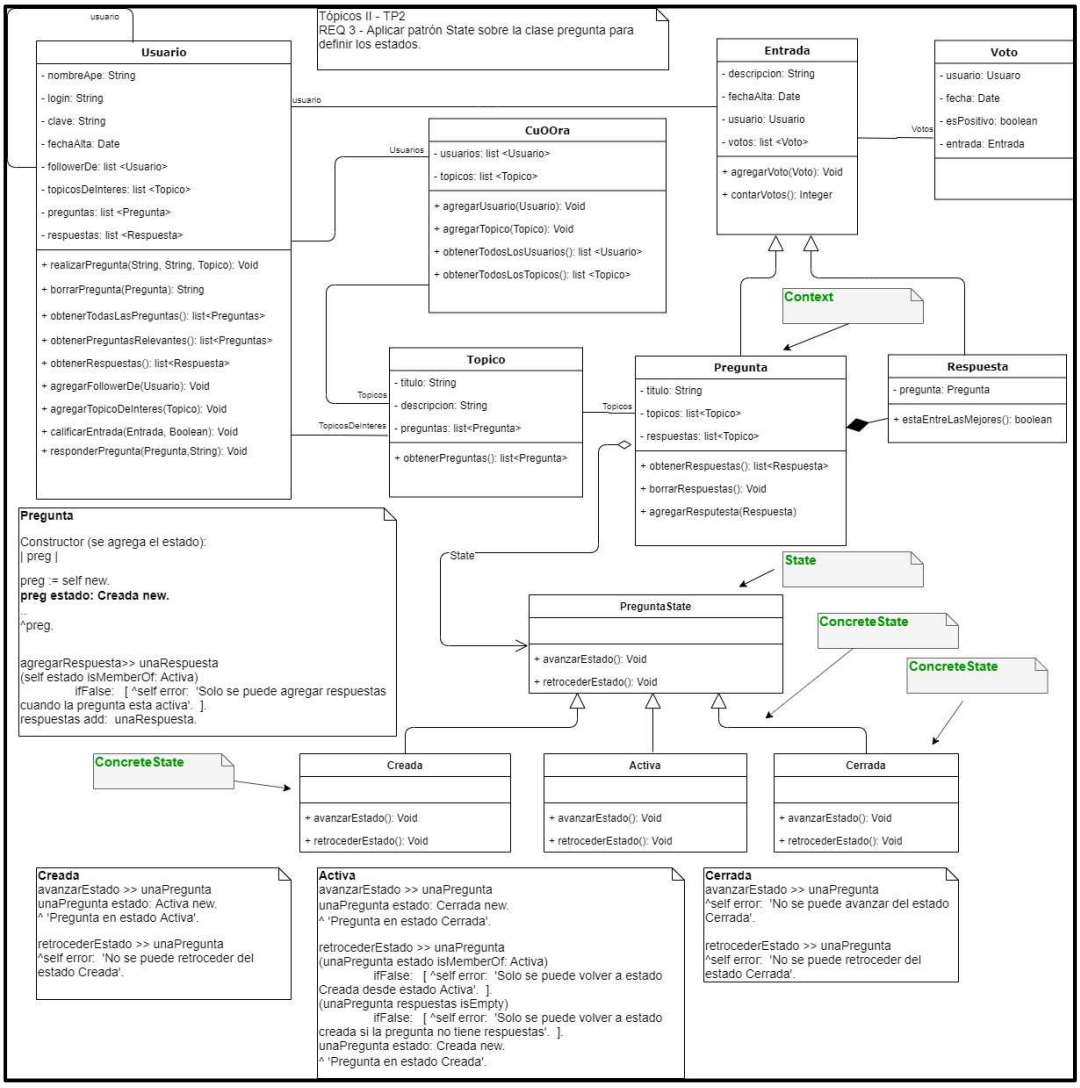
```
Constructor (se agrega el estado):
| preg |

preg := self new.
preg estado: Creada new.
..
```

```
..
^preg.
```

Modificación del método agregarRespuesta de clase Pregunta

```
agregarRespuesta>> unaRespuesta
(self estado isMemberOf: Activa)
                ifFalse:   [ ^self error:  'Solo se puede agregar respuestas cuando la pregunta esta activa'.  ].
respuestas add:  unaRespuesta.
```



Requerimiento3

Requerimiento 4: Optimización del acceso a la información.

Se utiliza el patrón Proxy para optimizar la carga de objetos.

Para ello se genera la clase abstracta **Subject** donde se declaran los métodos **obtenerRespuestas** y **borrarRespuestas** correspondientes a la clase **Pregunta**.

Además, se crea la clase **PreguntaProxy** que hereda de **Subject**, y se modifica la clase **Pregunta** para que herede ahora de **subject**

PreguntaProxy al instanciarse recibe un objeto de clase Pregunta (el objeto real) y lo guarda en la propiedad pregunta.

Por último, se modifica el método **obtenerPreguntas** de la clase **Tópico** para instanciar los objetos de la clase **PreguntaProxy**.

Métodos implementados

PreguntaProxy

```
obtenerRespuestas
^self pregunta obtenerRespuestas

borrarRespuestas
^self pregunta borrarRespuestas
```

Pregunta

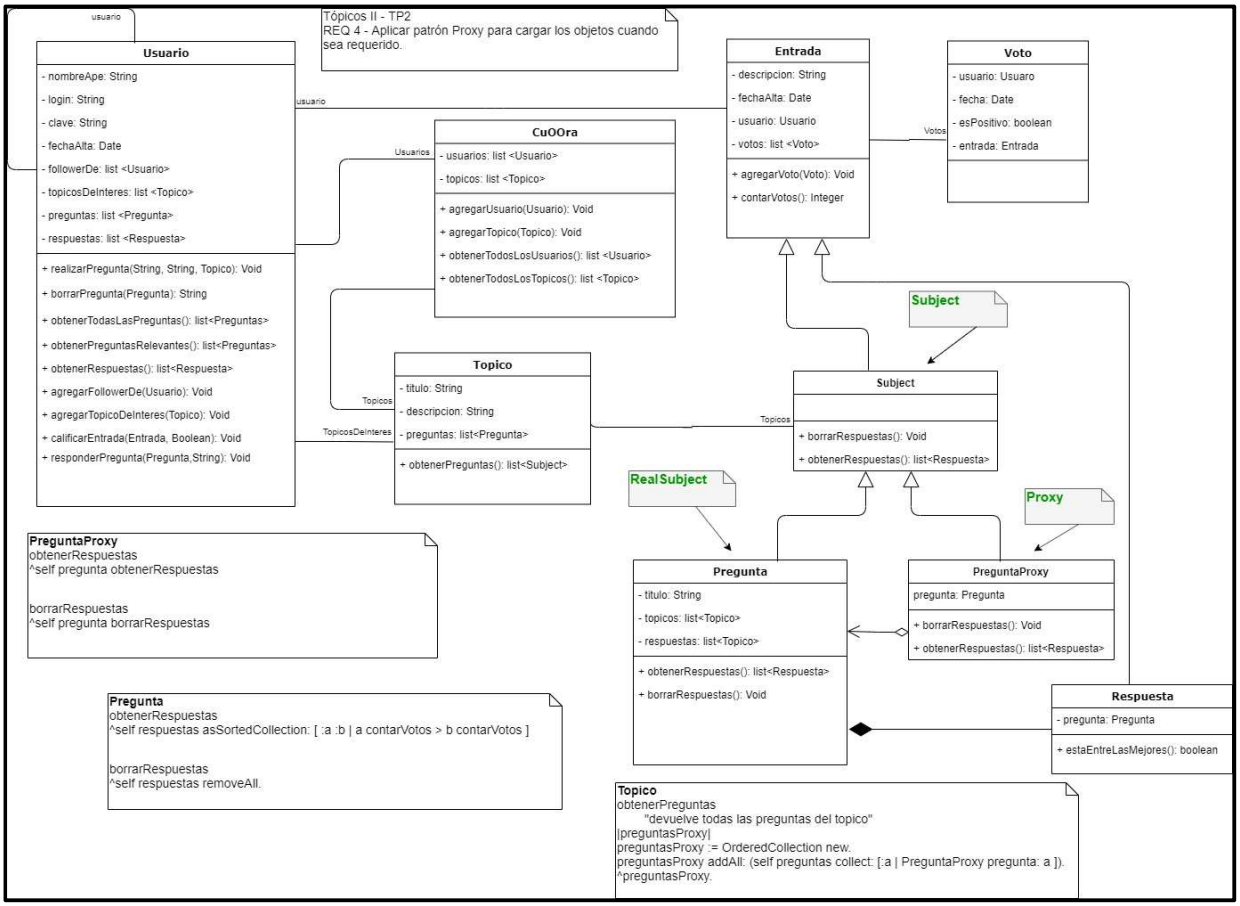
```
obtenerRespuestas
^self respuestas asSortedCollection: [ :a :b | a contarVotos > b contarVotos ]

borrarRespuestas
^self respuestas removeAll.
```

Modificación del método obtenerPreguntas de la clase tópico

Topico

```
obtenerPreguntas
        "devuelve todas las preguntas del topico"
|preguntasProxy|
preguntasProxy := OrderedCollection new.
preguntasProxy addAll: (self preguntas collect: [:a | PreguntaProxy pregunta: a ]).
^preguntasProxy.
```



Requerimiento 4