Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Ingeniería de Software II

TP 1 - Desarrollo Sprint y DOO

Grupo 9

Integrante	LU	Correo electrónico
Matías Nicolás Incem	396/09	matiasincem@gmail.com
Matías Ezequiel Barbeito	179/08	matiasbarbeito@gmail.com
Nicolás Alejandro Maffongelli	53/09	nicolas.maffongelli@gmail.com
Santiago Torres Batán	913/10	santi73@gmail.com

Contents

1 Introducción

2 User Stories

3 Diseño de la aplicación

3.1 Post

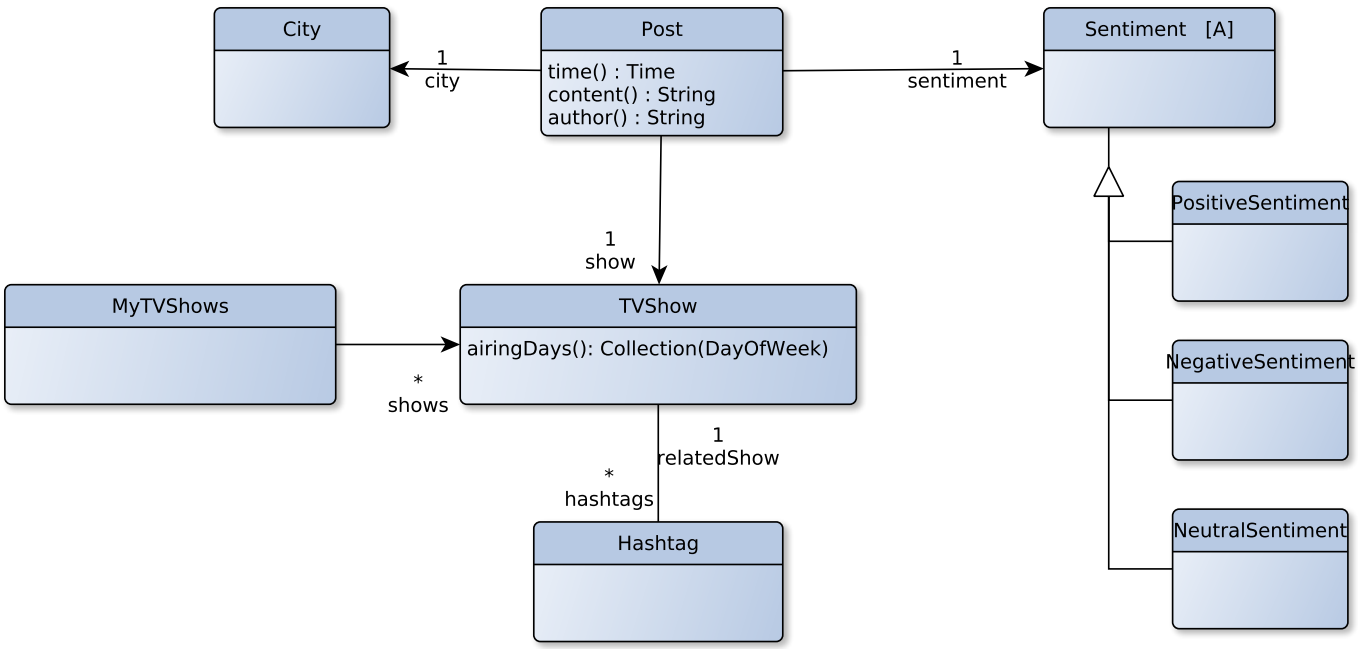

Figure 1: Post

El *post* es el elemento básico de esta aplicación, y consiste en un mensaje enviado a través de una red social. En principio es la representación de los datos de un *tweet* que son útiles en nuestro programa, y también puede representar estos mismos datos, obtenidos de una red social distinta de *Twitter*. Contiene algunos datos básicos como autor, texto del mensaje y fecha/hora de envío. También tiene un indicador de la ciudad desde la que se mandó el mensaje, en caso de que el autor haya activado la localización por GPS y de esa forma se pueda identificar su procedencia. Este indicador resulta útil para la vista de popularidad en mapa. Cada post también está vinculado con un único show de TV, el cual se le asigna basándose en palabras claves contenidas en el mensaje. Por último, posee un indicador de sentimiento, que lo clasifica como positivo, negativo o neutro.

3.2 Post Filterer

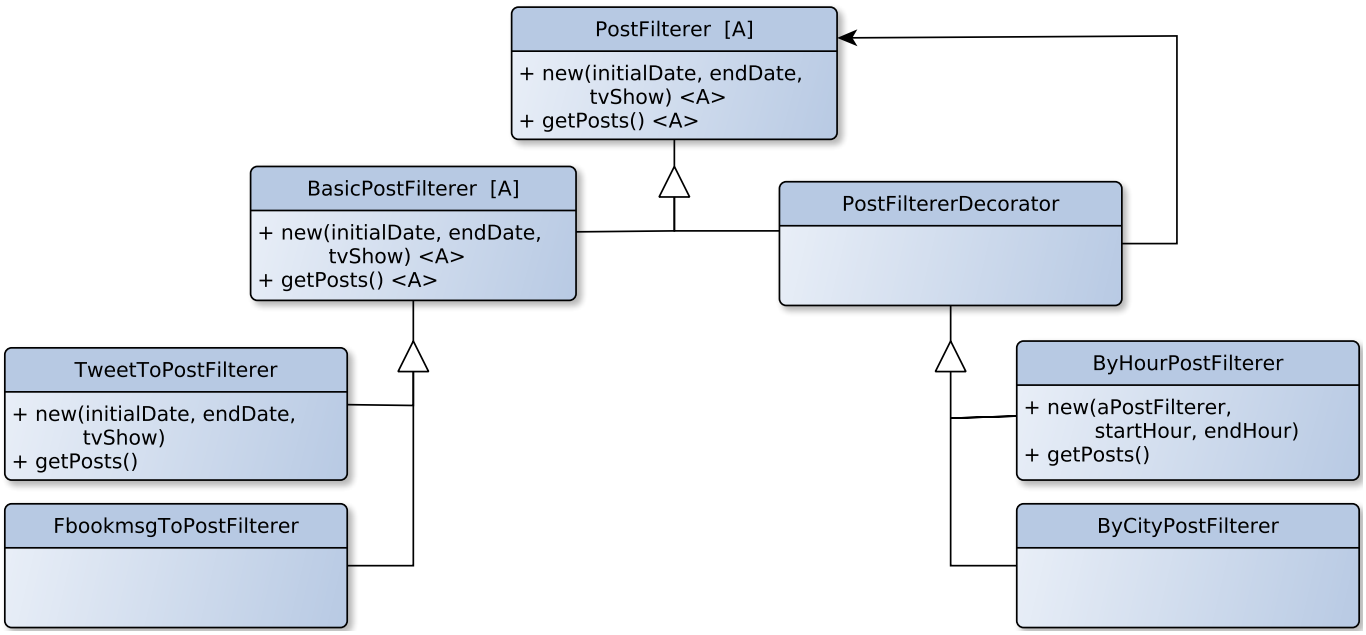

Figure 2: Post Filterer

3.3 Sentiment Analyisis

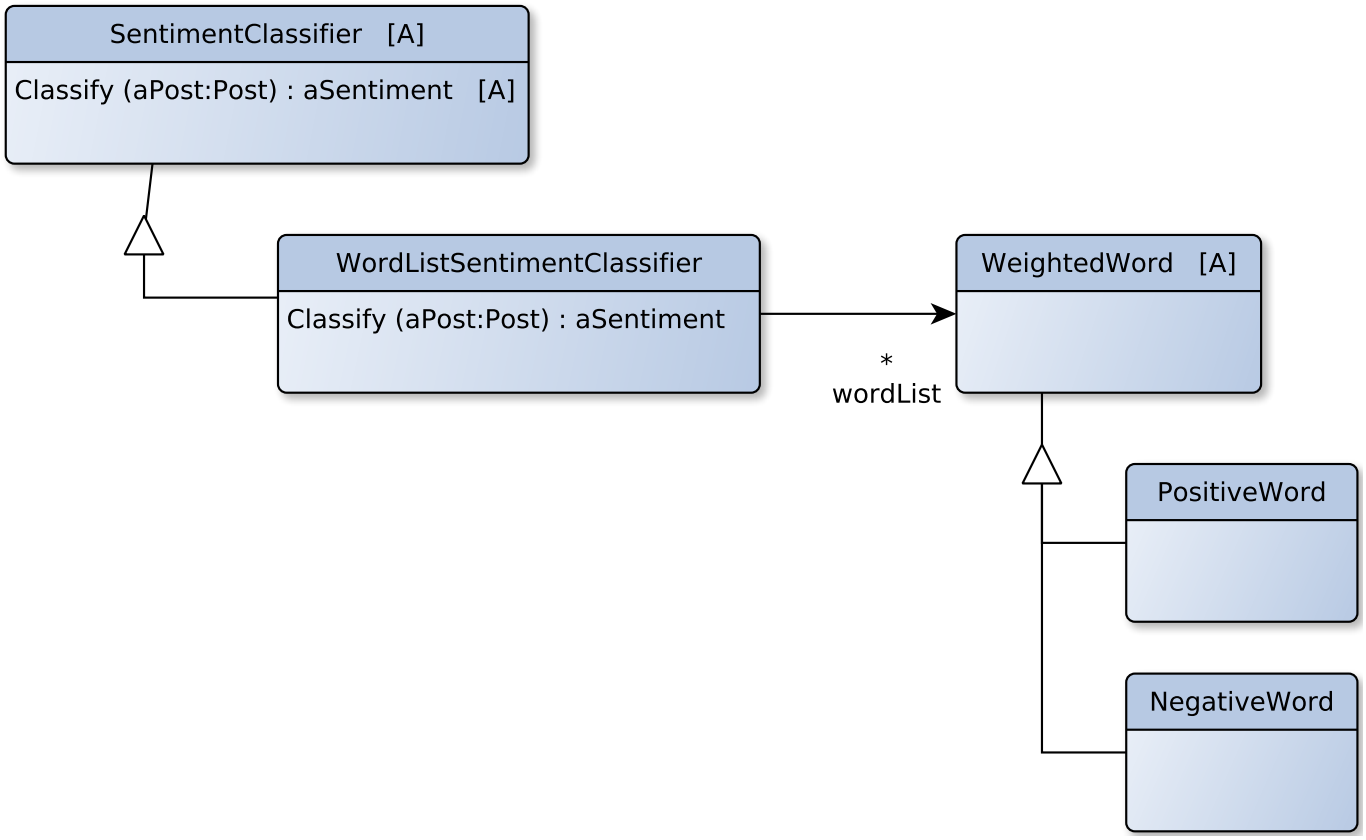

Figure 3: Sentiment Classifier

Este es un módulo de *Sentiment Analyisis*, que consiste de un clasificador que asigna un valor a cada post y es llamado luego de recibir los datos de la red social e integrarlos al modelo de la aplicación. Existen tres valores para el sentimiento: positivo, negativo y neutro, y la manera de clasificar los posts es transparente al resto de las funcionalidades. En esta primera versión del programa, se usa un clasificador simple que lee de un corpus de palabras puntuadas como positivas o negativas, y realiza la clasificación según la cantidad de palabras de cada tipo que aparecen

en cada mensaje. Se espera que a futuro se diseñe un nuevo clasificador con un criterio más complejo, y cuando esté implementado, se sustituya el que existe actualmente.

3.4 Meters & Views

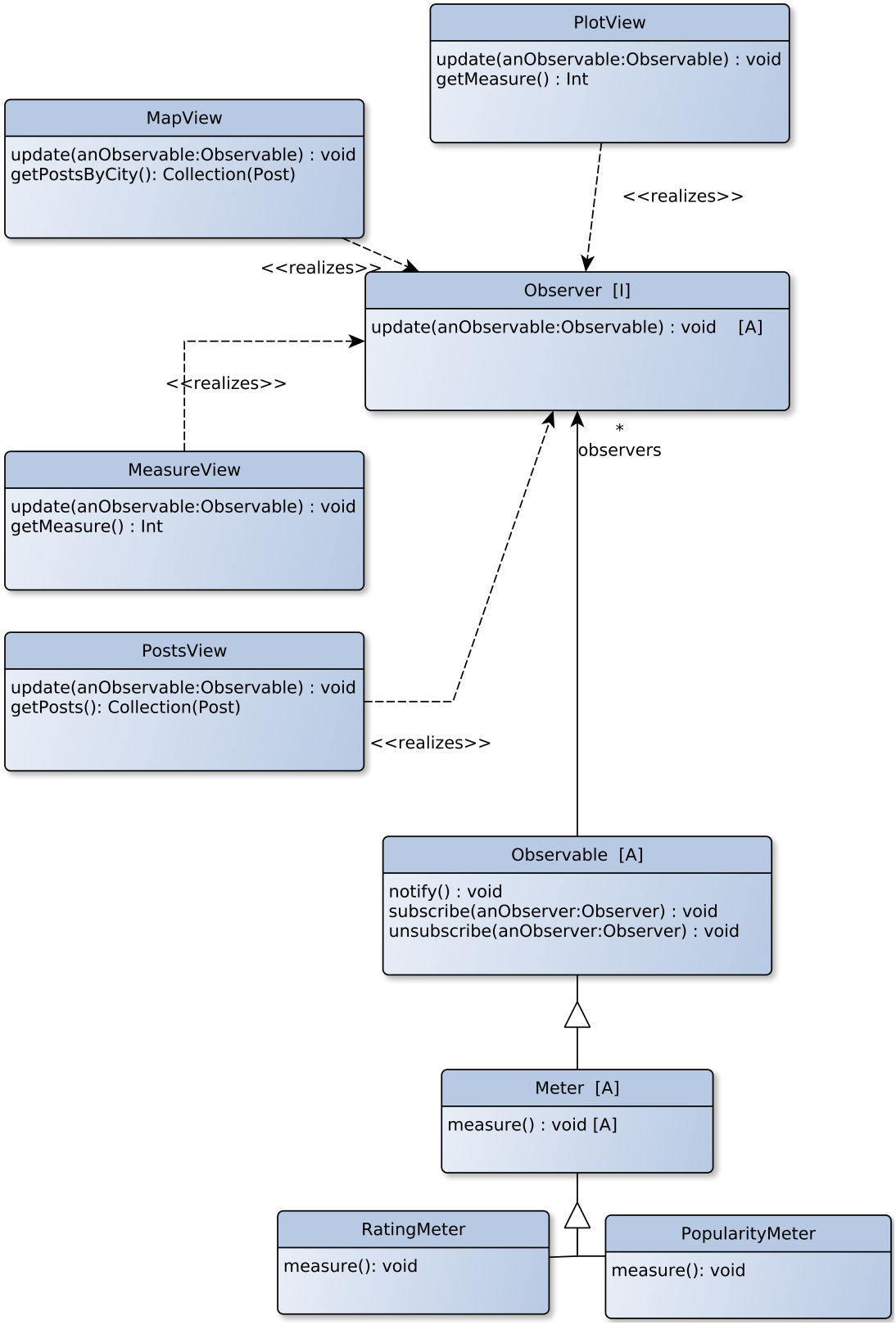

Figure 4: Meters & Views

Existen dos maneras de realizar mediciones en la aplicación: el rating y la popularidad. El rating consiste en una medición de la cantidad de posts recibidos para un programa en determinada fecha, durante el horario de emisión del programa, con un peso asignado según el sentimiento del mensaje. El rating puede consultarse "en vivo" para la fecha actual, mientras un programa se está emitiendo, y en ese caso se actualizará cada 10 segundos la información

obtenida. La popularidad es una medición de la cantidad de posts de un programa, dado un intervalo de tiempo, en cualquier horario, también pesados según sentimiento.

En la aplicación hay varias formas de visualizar esta información: la funcionalidad básica es la de mostrar un número que represente la medición de rating o popularidad (que aparece nombrada como MeasureView). Se puede pedir que el resultado se muestre en un gráfico. También hay una funcionalidad para poder leer los mensajes concretos que la aplicación obtuvo y usó para sus cálculos. Estos mensajes se muestran con autor, contenido y fecha, y se los puede filtrar de acuerdo a su sentimiento, en caso de que el usuario quiera ver u ocultar algún tipo de contenido. Por último está la opción de ver los posts en un mapa, donde se los clasifica por área geográfica y se muestra el valor de la medición para cada zona urbana principal.

Cada una de estas vistas es independiente de las demás y trabaja con información que puede obtener del medidor que le corresponde. Esto permite que se puedan agregar o modificar los tipos de vista existentes sin un gran impacto en el resto de la aplicación, y además hace que no sea necesario mantener activas las vistas que el usuario no solicitó.

3.5 Interfaz de usuario

La interfaz de usuario es el elemento que coordina y da inicio al nivel más alto de la aplicación. Su responsabilidad es permitir que el usuario active las funcionalidades que desea usar, y eso lleve a crear los objetos necesarios. Cada vez que el usuario seleccione un programa de TV de la lista, y pida ver el rating o la popularidad de dicho programa para un intervalo, se creará un medidor correspondiente a los datos recibidos. De forma similar, cada vez que se seleccione una nueva manera de visualizar los datos, la interfaz pedirá la creación de una vista vinculada al medidor, para que muestre la información.

4 Retrospectiva

Durante el último mes trabajamos en la primera iteración del ciclo iterativo incremental de desarrollo de la aplicación, según la metodología ágil de *SCRUM*. Analizando nuestro método de trabajo y los resultados obtenidos, llegamos a las siguientes conclusiones:

En primer lugar, nos resultó difícil adaptarnos a la metodología de trabajo ágil; hubo momentos de poca comunicación en el grupo, y en algunos casos no logramos hacer una distribución de tareas tal que cada miembro del grupo tuviera claro quién tenía asignada cada responsabilidad. Además no logramos organizar las tareas en un primer momento, por lo que una primera parte del *sprint* no fue aprovechada. Combinado con algunas obligaciones académicas de los miembros del grupo, esto resultó en que la cantidad de trabajo que realizamos fue aumentando a medida que el final del sprint se acercaba, en ves de ser relativamente constante como hubiera sido ideal.

Por otro lado destacamos que, cuando organizamos una división de tareas con reuniones o charlas periódicas para mantenernos informados, pudimos avanzar más rápido con las tareas, en especial el desarrollo del código fuente, cada uno trabajando en módulos encapsulados.

También llegamos a la conclusión de que es necesario tener claro el diseño de la aplicación para poder escribir el código fuente. En un principio teníamos algunos módulos del proyecto mejor diseñados que otros, y nos ocurrió que, en los que tenían un diseño más vago o poco claro, encontrábamos objetos que parecían estar aislados del modelo, sin usarse, o atribuciones que estaban repetidas entre varios objetos. En algunos casos fue necesario reunirse entre los miembros del grupo para desambiguar ese tipo de situaciones, lo que nos forzó a detallar más el diseño de esa parte para poder entender el funcionamiento que estábamos imaginando. Creemos que para el futuro es importante tener un diseño global hecho, definir las interfaces o uniones entre módulos que sean relativamente estables (poco volátiles) y basar las siguientes decisiones en ese diseño.