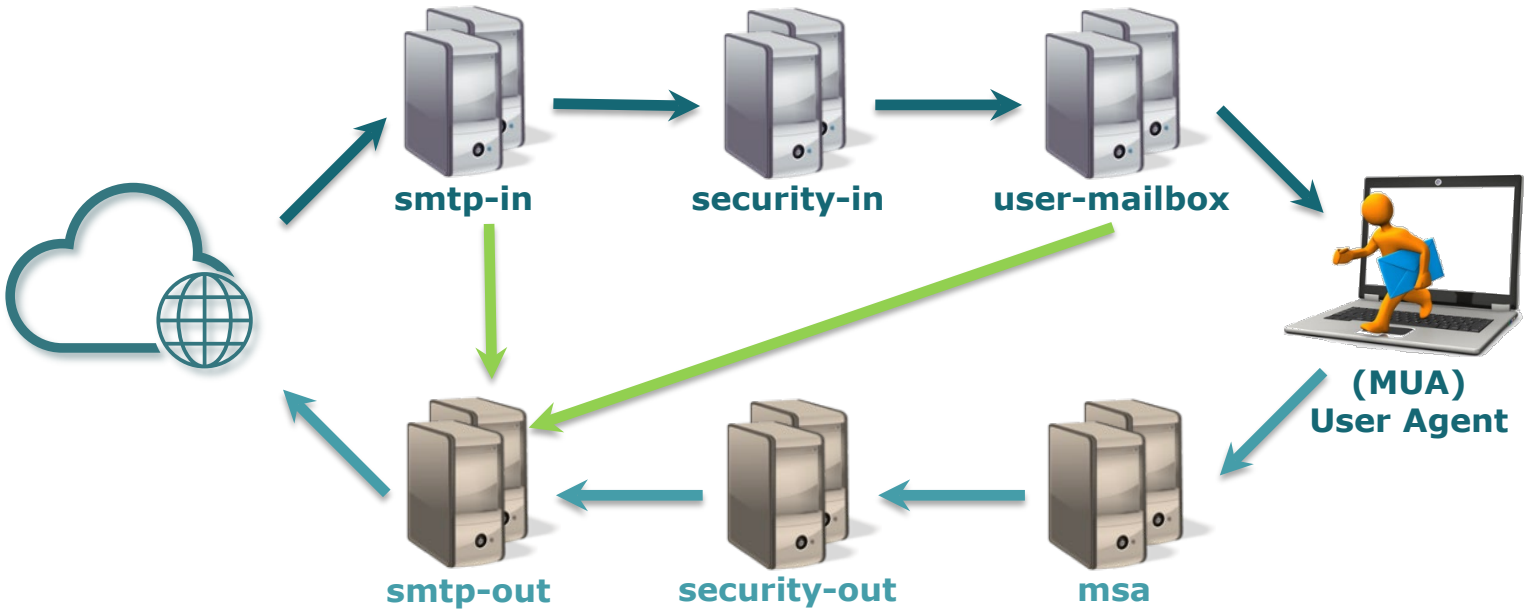
Expresiones regulares

Práctica 2

Arquitectura



Ejemplo de traza


```
2020-02-18 12:01:18 security-out2 [FB700FBB]: connect from: msa1.etsist.upm.es
2020-02-18 12:01:18 security-out2 [FB700FC5]: connect from: msa1.etsist.upm.es
2020-02-18 12:01:18 security-out2 [FB700FBB]: message from: <ricardo.salvador@etsist.upm.es> to:
<gregorio.santos@carapa.cz> message-id: <61798569@maroger.etsist.upm.es> size: 5525
2020-02-18 12:01:18 security-out2 [FB700FC5]: message from: <paco.caballero@etsist.upm.es> to:
<lorenzo.espinosa@cocodrilo.ceiba.cm> message-id: <57343771@giorgione.etsist.upm.es> size: 8511
```

Paso 0. Crear y verificar las expresiones regulares

Antes de programar es conveniente construir y verificar las expresiones regulares usando alguna de las siguientes herramientas online:

`https://www.rubular.com`
`https://regex101.com`

usando como entrada una línea de texto de los ficheros de log

Paso 0. Crear un programa simple

Clase **EjemeploSimple**:

- A partir del código disponible en https://docs.oracle.com/javase/tutorial/essential/regex/test_harness.html o el generado en regex101 realizar un pequeño programa que:
  1. Establezca la expresión regular (ya hecho).
  2. Abra uno de los ficheros proporcionados para la práctica.
  3. Para todas y cada una de las líneas del texto del fichero:
     1. Lea la línea y genere el objeto de la clase Matcher que permita procesar el resultado.
     2. Haga cualquier procesado si el match ha sido correcto (contar el número de matches correctos, escribir la línea por pantalla, ….

Paso 1. Codificar la aplicación

Clase **EstadisticasLog**:

- Clase inicial que recibe como parámetro el directorio donde están los ficheros de log
- Verifica que recibe el argumento del directorio y que este es válido
- Crea las colecciones que se usarán en la aplicación
- Crea un objeto Pattern con un patrón de traza bien formada y con los grupos necesarios para extraer la información
- Crea los hilos trabajadores a los que le pasa como argumentos el fichero de log a procesar, las colecciones y el patrón de traza
- Espera a que terminen los hilos
- Muestra la información de las colecciones por la salida estándar

Paso 1. Codificar la aplicación

Clase **Trabajador**:

- Clase que procesa el fichero log que recibe como parámetro
- Por cada línea de texto del fichero:
  - Verifica que está bien formada
  - Guarda en las colecciones correspondientes la información que extrae de la línea

Importante: cada fichero solo puede ser procesado una vez

Ejemplo de expresión regular

2020-02-20 16:22:57 msa1 [45E97DB9]: connect from: benton.etsist.upm.es

```
[0-9]{4}-[0-9]{2}-[0-9]{2}

\s[0-9]{2}:[0-9]{2}:[0-9]{2}

\s[^0-9]+[0-9]+

\s\[\w+\]:.*
```

Hay un error (depurar)

Para un mensaje de entrada:

```
^[0-9]{4}-[0-9]{2}-[0-9]{2}\s+[0-9]{2}:[0-9]{2}:[0-9]{2}
\s+smtp-in[0-9]+\s+.*\sconnect from:\s*([^\s]+).*
```

Grupos

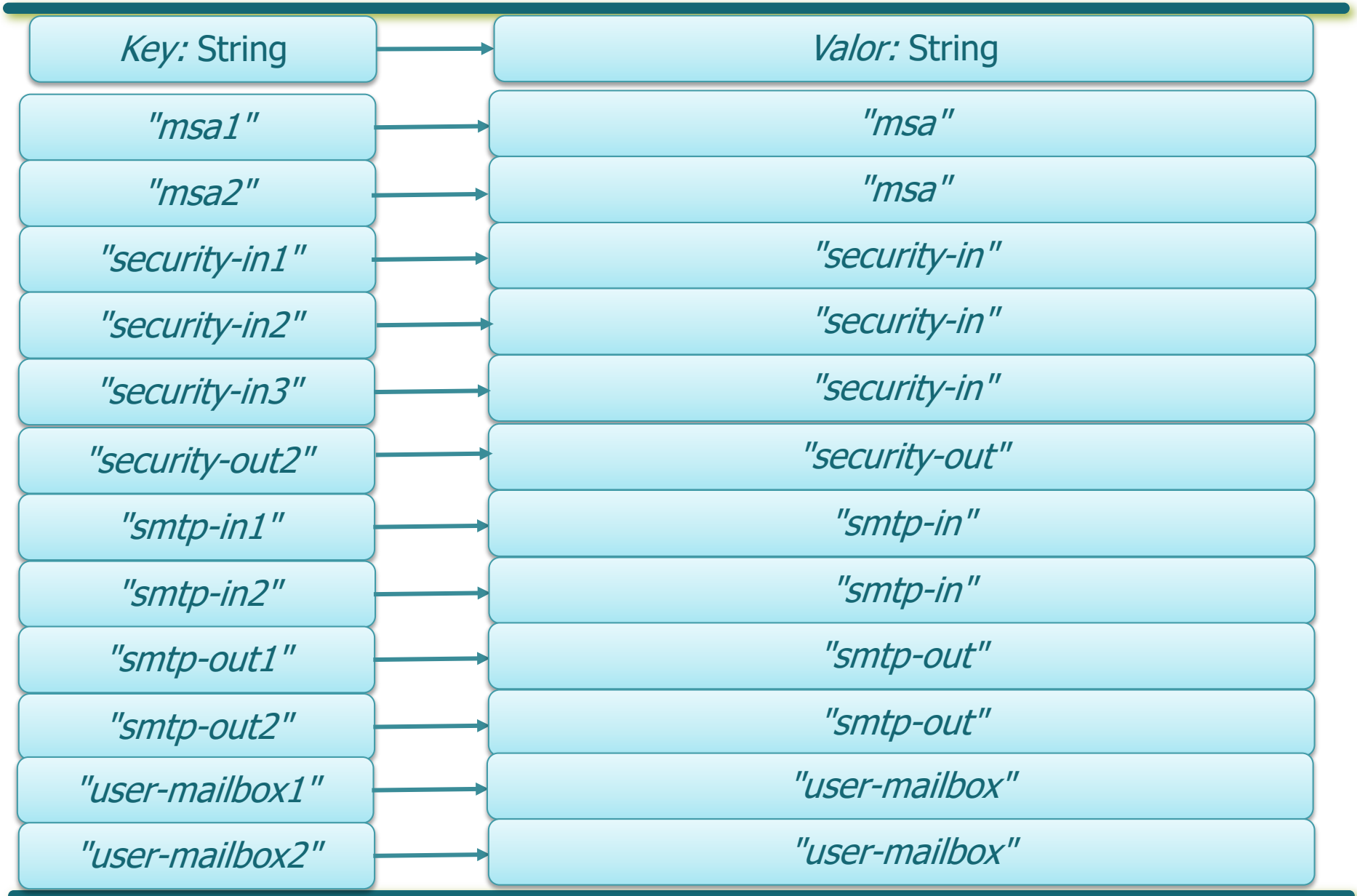
```
([0-9]{4}-[0-9]{2}-[0-9]{2})\s([0-9]{2}:[0-9]{2}:[0-9]{2})\s[^0-9]+[0-9]+
```

```
Matcher m;
String lineaTexto
…
….
Pattern pattern = Pattern.compile(…);
……
……
m = pattern.matcher(lineaTexto);
if (m.matches() ){
```

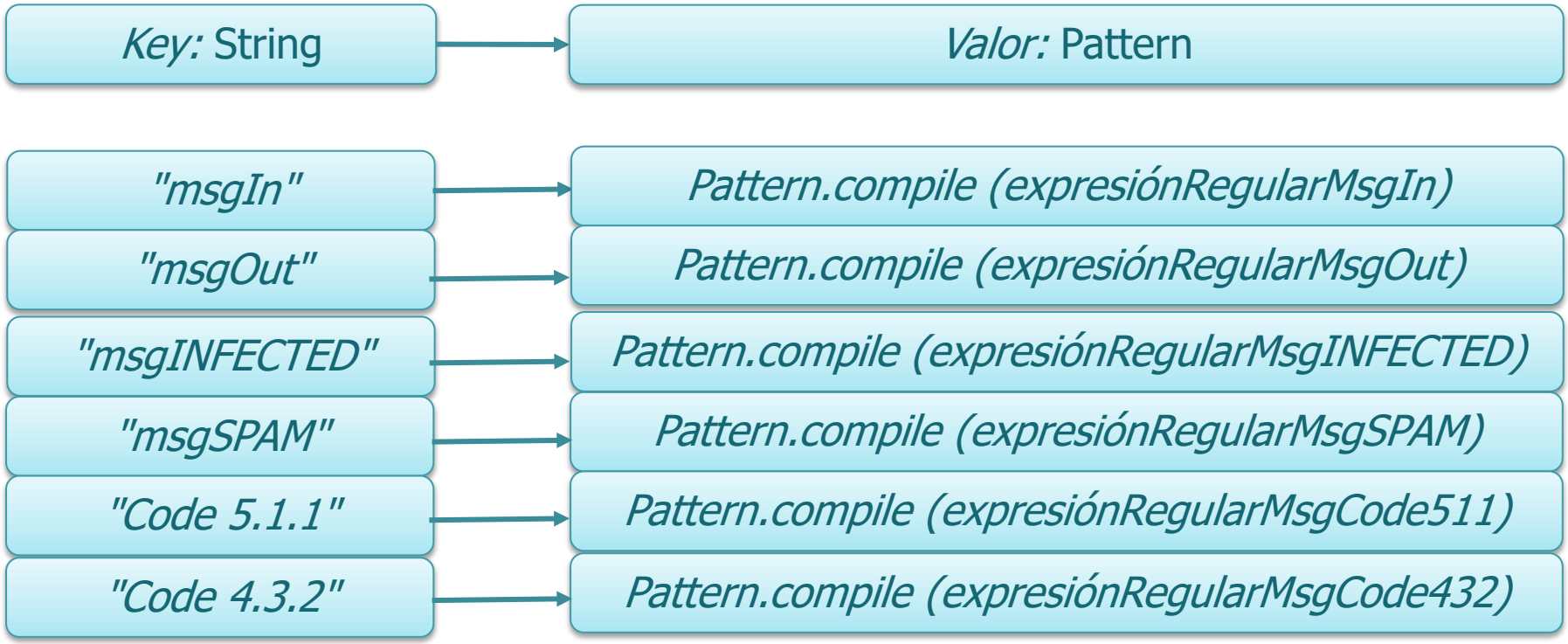
m.group(1) → la parte de lineaTexto conforme a [0-9]{4}-[0-9]{2}-[0-9]{2}

m.group(2) → la parte de lineaTexto conforme a [0-9]{2}-[0-9]{2}-[0-9]{2}

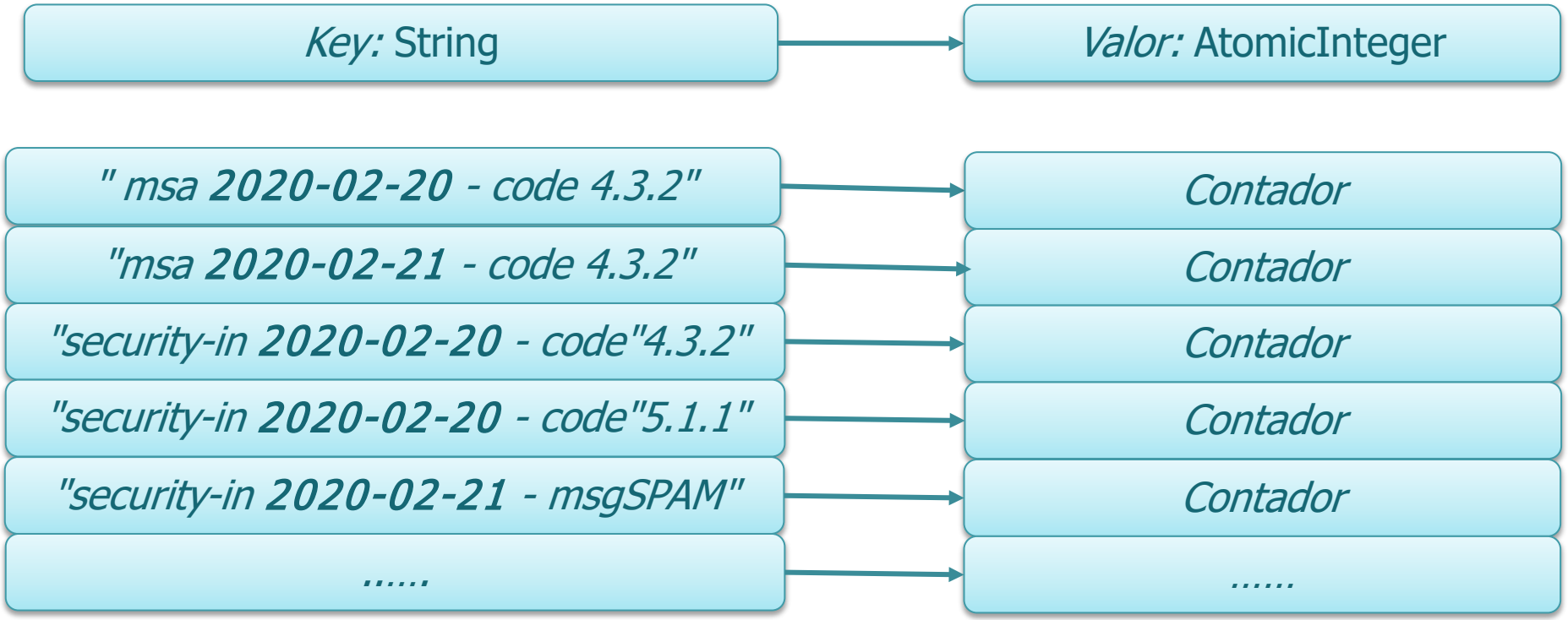
Mapa *hmServidores*: almacena el nombre y tipo de servidor



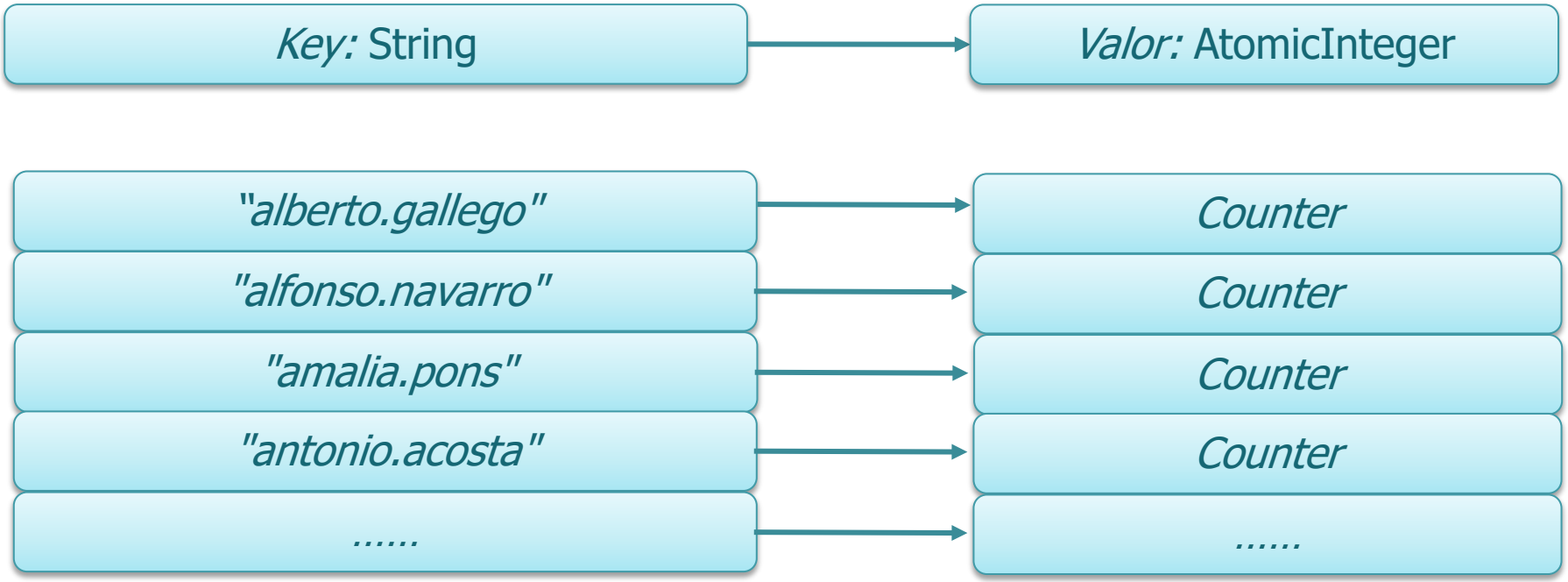
Mapa hmPatronesEstadisticasAgregadas: almacena las expresiones regulares



Mapa hmEstadisticasAgregadas: almacena las ocurrencias encontradas de las estadísticas agregadas



Mapa hmUsuarios: almacena el nº de mensajes enviados por cada usuario



Expresiones regulares

Práctica 2