**Trabajo Práctico N⁰ 2.3**

**Técnicas de Diseño**

**Grupo 10**

**Integrantes:**

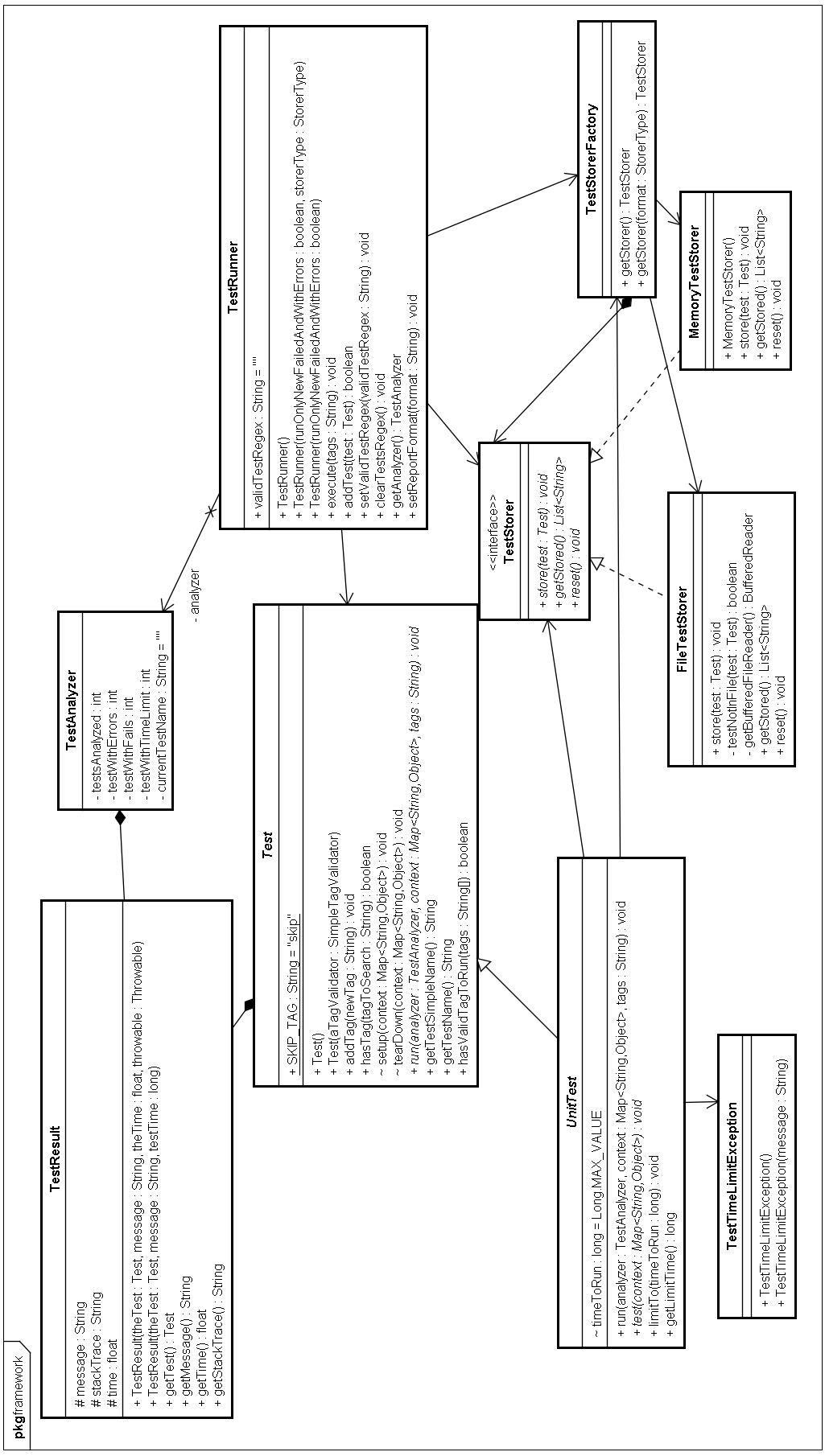
81052 – Cesar Leguizamon ([fabi1816@gmail.com](mailto:david.mdq89@gmail.com))

90110 – David Marcos ([david.mdq89@gmail.com](mailto:david.mdq89@gmail.com))

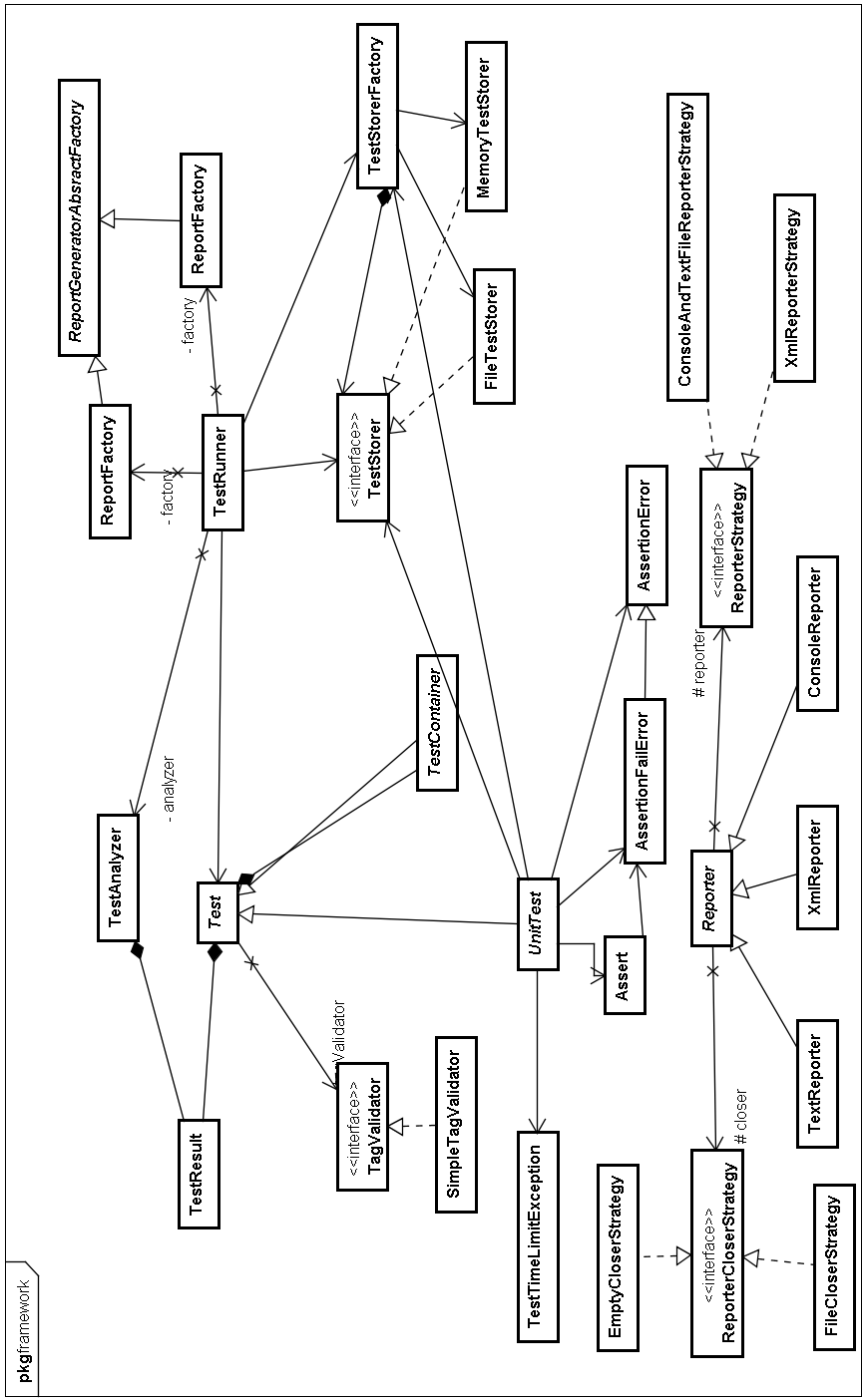
91299 – Diego Meller ([mellerster@gmail.com](mailto:mellerster@gmail.com))

**Diagrama de clases**

Estos fueron los cambios que hemos agregado para adaptar a la nueva entrega:



A continuación se puede observar el diagrama de clases completo:



**Críticas del trabajo del grupo anterior**

**Diseño**

Han utilizado varios patrones, con lo cual facilito mucho el trabajo de poder entender la resolución del trabajo práctico.

Los patrones que han utilizado son:

* Container Pattern: La clase TestContainer es una clase contenedora de la interfaz Test.
* Template Pattern: La clase abstracta UnitTest le da una implementación al método run() de la interfaz Test.
* Strategy Pattern: Hay varios strategy como por ejemplo la estrategia para generar un reporte.
* Builder Pattern: Se usa este patrón para generar las diferentes partes que se asocian con un reporte.

**Código**

El código está muy bien detallado, utilizan un único idioma correctamente. Al principio nos costó encontrar como funcionaba el programa ya que no había un paquete de utilización previo pero luego de investigar más detalladamente el código lo hemos encontrado sin problemas.

**Test**

La herramienta es sencilla para ejecutar muchas pruebas al mismo tiempo. Para testear algo se deben seguir los siguientes pasos: crear una instancia de Unit Test, sobreescribir el método test, y luego para evaluar algún valor se deben utilizar los Asserts provistos por el framework.

Además han realizada una gran cantidad de pruebas, las cuales testeaban gran parte del desarrollo.

**Herramientas utilizadas**