**Nombre Alumno: Sena Nicolás Ariel**

**LU:**

**Nombre Profesor:** Burghardt – Mettini

**Grupo Laboratorio:** Grupo 3

**TP:** 3

**Fecha:** 14/09/17

Luego de aprender los conceptos básicos de este paradigma orientado a objetos nos metemos más de lleno en el manejo de clases, específicamente en la forma de conocimiento entre objetos.

Vimos de qué manera los objetos se envían mensajes, donde el destinatario responde con un método implementado en la clase al que pertenece. Pero para poder responder ese mensaje el objeto debe conocer al objeto que envía ese mensaje, es decir nombrarlo, que exista una ligadura. Como por ejemplo cuando instanciamos el objeto “unaCuenta” de clase “Cliente” y la ligamos a un nombre que hace referencia, “titular”

Existen distintos tipos de conocimiento:

Variables de instancia: La relación se da entre un objeto y sus atributos, por lo tanto dicha relación nace cuando se instancia el objeto y muere junto con el. Para crear tal relación debemos instanciar un objeto “B” y pasarle como constante a un objeto “A”, este objeto tendrá la referencia en sus atributos y A podrá enviarle mensajes a B, siempre y cuando posea el método adecuado para responder.

Parametros: La relación de conocimiento se da dentro del cuerpo de un método, un objeto C al mandar el mensaje a un objeto A envía como parámetro a otro para que el receptor pueda cumplir el requerimiento.

Variables temporales: se utilizan para nombrar objetos dentro de un método cuando sea necesario utilizarlo de manera auxiliar, la relación se establece desde la referencia establecida hasta lo que dure la activación del método.

Seudo Variables: Es la forma que tiene un objeto de conocerse a si mismo, funciona manera similar a que una variable de instancia, con la diferencia de que no es un atributo de ninguna clase y no se le puede asignar un valor, en Java utilizamos “this” como pseudo variable para nombrarse un objeto a si mismo.

En un modelo UML existen 4 tipos de relaciones entre sus elementos: realización, generalizacion y las explicadas a continuación.

Asociación: Cuando una clase tiene en su estructura a otra clase, se representa con línea continua y a veces etiquetas. Pueden ser bidireccionales o unidireccionales, dependiendo de la restricción que se le quiera dar, en la línea ubicamos el nombre de referencia para conocer al otro objeto, y un numero para representar su multiplicidad.

Dependencia: Se establece con líneas punteadas, que parte desde la entidad que requiere un servicio y apunta hacia otra entidad que va a proveer el servicio solicitado. Es importante que el cliente no conozca los detalles internos del servidor para evitar cambios en cascada, la relación utiliza variables temporales.