UNSE
Universidad Nacional de Santiago del Estero

REVISIÓN DE ALTERNATIVAS DE ADAPTACIÓN DE UN SISTEMA

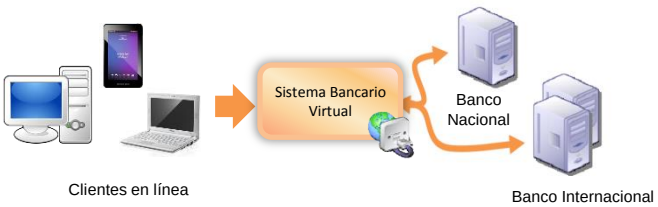
Del Corro, Gonzalo Gabriel

Instituto de Investigación en Informática y Sistemas de Información – Facultad de Ciencias Exactas y Tecnologías

INTRODUCCION: Los cambios que se producen en el entorno de las organizaciones hacen que sea necesario que el software que utilizan para realizar sus actividades, acompañe esos cambios y se adapte a los nuevos requerimientos para seguir siendo útil y mantener la satisfacción de los usuarios.

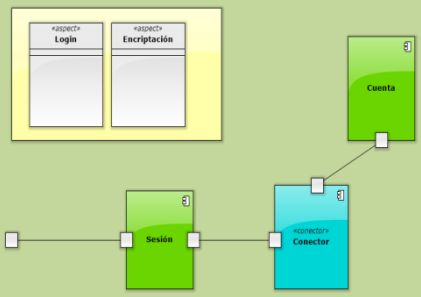
OBJETIVO GENERAL: Comprender cómo se logra la adaptación de los sistemas web empleando distintos paradigmas de programación.

Caso de Estudio:
Transferencia de dinero en un sistema bancario virtual



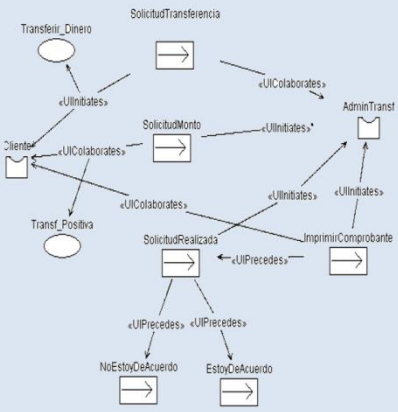
PROGRAMACIÓN ORIENTADA A ASPECTOS

La POA permite encapsular los intereses que "atraviesan" el sistema, en entidades bien definidas llamadas *aspectos.* Un aspecto es una combinación de "puntos de unión", "puntos de corte" y "avisos" .
A efectos de seguridad, se debe establecer un mecanismo de login y encriptación para realizar la transacción. Ambos asuntos, requieren de adaptación dinámica por variaciones de comportamiento sobre el flujo de control ya que, por cada sesión bancaria que se inicie, se debe crear dinámicamente una instancia de esa sesión para asegurar que las operaciones sean seguras, fiables y confidenciales.



PROGRAMACIÓN ORIENTADA A AGENTES

El sistema puede concebirse como un sistema multiagente. Debido a la falta de conocimiento de las entidades que pueden aparecer y desaparecer en el contexto en algún momento determinado, es necesario tomar decisiones acerca del acceso.
Cuando un *AgenteCliente* o *AgenteBanco* accede al sistema, se generan nuevas dependencias a las que el modelo de coordinación debe responder a partir de los agentes presentes en el entorno.



PROGRAMACIÓN ORIENTADA AL CONTEXTO

El loggin y la encriptación se corresponden con intereses cruzados y, por lo tanto, ambos pueden encapsularse en dos capas *EncriptacionCapa* y *LogginCapa* dentro de la clase *Cuenta*. Ambas capas contienen definiciones de métodos parciales que pueden sobrescribir la definición del método base cuando la capa se active, lo cual permite diferentes estrategias de proceder para sus correspondientes métodos que, en el ejemplo, van a depender de las distintas políticas de seguridad bancaria establecidas para cada uno de los clientes.

```
public class Cuenta{
    ...
    layer EncriptacionCapa{
        public void credito(int monto){
            proceed (Encriptacion.desencriptar(monto));
        }
        public void debito(int monto){
            proceed (Encriptacion.desencriptar(monto));
        }
        public int getBalance(){
            return Encriptacion.encrptar(proceed());
        }
    }
    layer LogginCapa{
        after public void credito(int monto){
            Logger.logCredito(this, monto);
        }
        after public void debito(){
            Logger.logDebito(this,monto);
        }
        public int getBalance(){
            int balance = proceed();
            Logger.logSolicitudBalance(this, balance);
            return balance;
        }
    }
}
```

Conclusión: La relevancia de este trabajo radica en la visión integradora que proporciona en el campo. En trabajos futuros se continuará comparando y analizando las fortalezas y debilidades, así como los distintos casos o ámbitos convenientes para la utilización de cada una de estas alternativas.

El trabajo fue realizado en el marco de la Beca CIN Convocatoria 2013 – 2014
dentro del Proyecto Acreditado "Métodos, Técnicas y Herramientas para la Ingeniería de Software Orientada a Agentes"