Fwk.Blocking

BLOCKING

El sistema de blocking que provee el Framework de Fwk permite a las aplicaciones (Back-End o Front-End) disponer un conjunto de mecanismos que facilitan el bloqueo pesimista de tablas o conjunto de tablas con información concurrente critica en cualquier sistema.

Básicamente el modelo de blocking consiste en establecer marcas de bloque en una base de datos. Estas marcas identifican una tabla de la base de datos del sistema transaccional, el atributo y valor de la tabla por el que se intenta bloquear.

Arquitectura

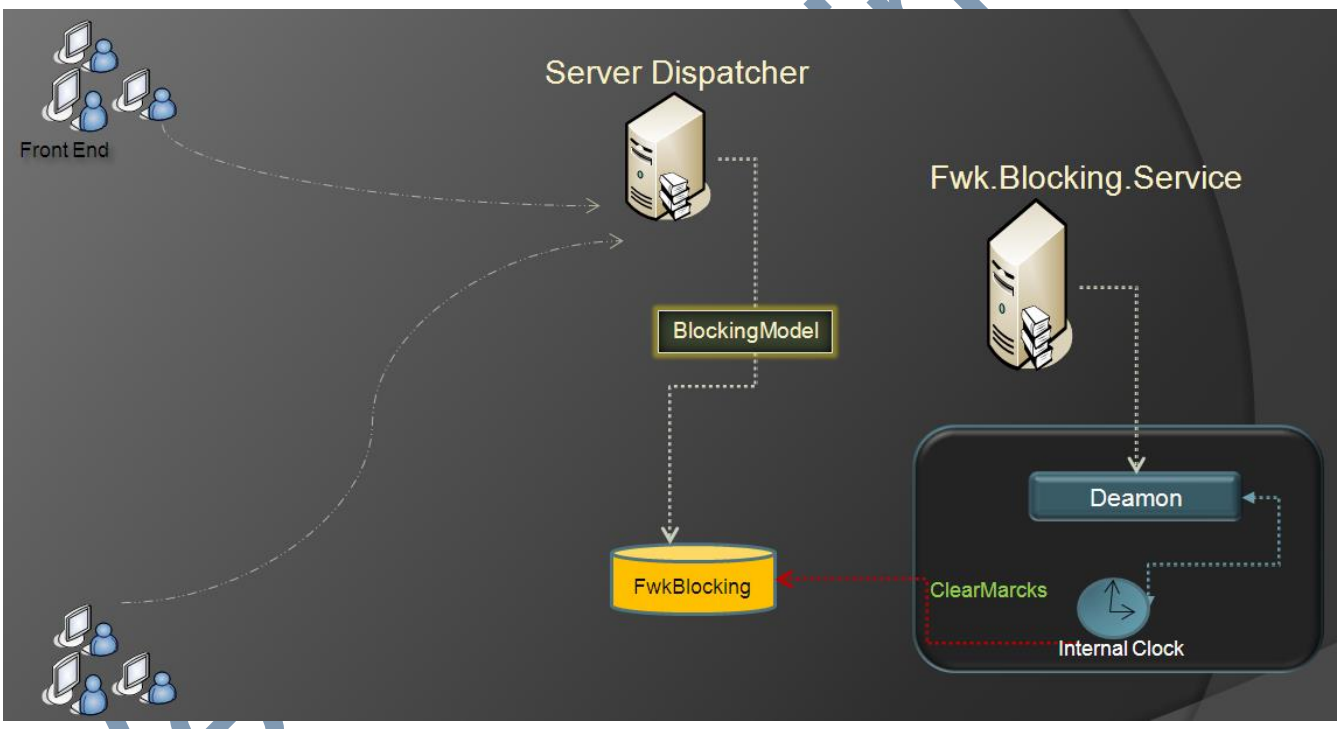

Figura 1.0

Servicio de Blocking

Se trata de un simple servicio de Windows demonio que con una frecuencia configurable determina si se deben remover las marcas de bloqueo de la base de datos FwkBlocking.

Para determinar si una marca caduco o alcanzo su tiempo de expiración, el servicio chequea el valor DueTime de la marca de bloqueo. Si **DueTime** < Now la marca expiro y es eliminada.

***Nota**: No hay que preocuparse por generar el valor DueTime. Este atributo se autogenera con la inserción de una marca de bloqueo a través del Engine de blocking.*

Dispatcher

El dispatcher o servidor de aplicaciones también tiene acceso a la base de datos FwkBlocking haciendo uso del modelo de blocking explicado más abajo. Su acceso es obviamente necesario ya que es el despachador de servicio a través de los diferentes servicios en un contexto orientado a bloqueos quien genera las marcas de bloqueo, las consulta y eventualmente las elimina.

Modelo de componentes

Dentro del frameworck podemos identificar la librería que da soporte al modelo de blocking, esta es:

Fwk.Blocking.Blocking

Gráficamente contiene los siguientes componentes:

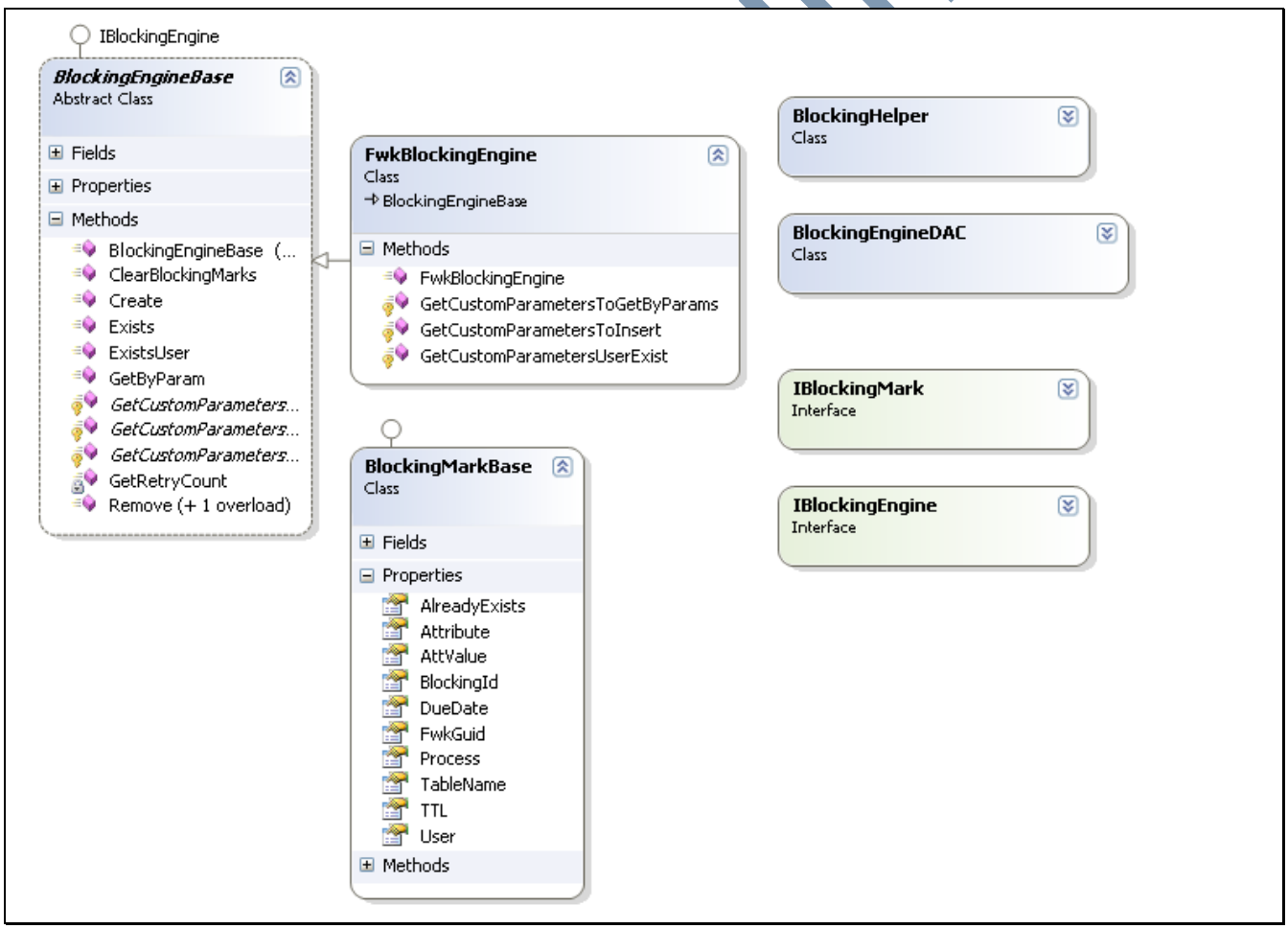

Figura 2.0

IBlockingMark Es la interfaz base de las marcas de bloqueo. Esta reprecenta los atributos de la tabla base en la base de datos de blocking.

Cualquier marca de bloqueo customizada debe implementar al menos esta interfaz.

BlockingMarkBase El la entidad base de las marcas de bloqueo. Reprecenta directamente la los atribiutos de la tabla de blocking en la Base de datos de bloqueng.

IBlockingEngine**:** es la interfaz de que de todos los engines de blocking. Tanto el engine básico como los customizados por los desarrolladores deben implementar esta interfaz a través de la clase base BlockingEngineBase

BlockingEngineBase Se trata de una clase abstracta que define el comportamiento basico de un manipulador de de marcas de bloqueo. Esta clase implementa los metodos generales y basicos para: crear eliminar buscar exitosamente marcas customizadas o estándar de blocking.

Métodos:

Nombre de método	Descripción
Create	Realiza un bloqueo creando una **BlockingMark**
Remove	Libera un contexto de bloqueo, borrando todas las marcas que posean el mismo id de Contexto.
ClearBlockingMarks	Elimina todas las marcas de bloqueo que hayan expirado. En otras palabras, limpia todas las marcas para las cuales se cumplió el TTL.(time-to-live)
GetByParam	Obtiene todas las marcas de bloqueo por parámetros. Es decir por los atributos de la clase que implemente `IBlockingMark`
Exists	Determina si existe una marca de bloqueo por su GUID o `BlockingId` `Retorna un true o false`
ExistsUser	Es similar a la `GetByParam pero no retorna las marcas sino el/los usuarios que tienen tomada esa marca de bloqueo.-`

Métodos abstractos:

Dispone de métodos abstractos que deben implementar las clases que hereden de esta base.

La clase que herede de BlockingEngineBase deberá implementar esto métodos para generar los parámetros que son enviados a una sentencia de creación, búsqueda o lo que sea necesario para adaptarse a los campos de una tabla customizada de marcas (que implemente IBlockingMark)

Nombre de método	Descripción

GetCustomParametersToInsert	Parámetros para insert
GetCustomParametersToGetByParams	Parámetros para búsqueda por varios campos de una tabla
GetCustomParametersUserExist	Parámetros para la llamada al método que busca marcas y retorna el usuario o los usuarios q la tienen tomada.

BlockingEngineDAC

Proporciona el acceso a datos del sistema de blocking.

FwkBlockingEngine

Esta es una clase por defecto que ya esta implementada eredando de la clase base BlockingEngineBase. Es uan clase que ya provee el framework de blocking a fines de interactuar con la tabla de la base de datos, tambien por defecto que ya esta generada.

Esta tabla se llama: **FwkBlocking**

Ejemplo demo:

A modo de que sea un poco más comprensible el modelo vamos a ver un simple ejemplo de utilización de Fwk.Blocking.

Supongamos que tenemos un sistema que consulta la tabla de productos para realizar modificaciones. Por lo tanto cuando un usuario trae la colección de productos selecciona uno y debe marcarlo para que nadie pueda hacer lo mismo.

Entonces al momento de obtener los datos del Producto se debería generar una marca de bloqueo al sistema de blocking.

El modo o los atributos por los cuales se bloquearan entidades es responsabilidad de diseño. Por ejemplo en este caso podríamos bloquear por tabla Producto, más ProductoID y el valor de este mismo.

Entonces hay que generar una marca de bloqueo utilizando la clase **BlockingMarkBase** ya que no es necesario desarrollar una **BlockingMark-Customizada** dado que la que existe en el framework de blocking responde a esta necesidad.

Nuestro servicio obtencion del producto al momento de generar una marca deberia antes preguntar si no esta siendo tomado por algun otro usuario.

El codigo se veria como sigue:

```
//(1)
BlockingMarkBase wMarkProducts = new BlockingMarkBase(null,"Productos");
//(2)
wMarkProducts.Process = ArchitectureTest.Common.ProcessEnum.Production.ToString();
wMarkProducts.Attribute = "ProductID";
wMarkProducts.AttValue = 100;

//(3)
FwkBlockingEngine wEngine = new FwkBlockingEngine();


List<string> wUserList = wEngine.ExistsUser(markProducts);
if (wUserList.Count != 0)
  {
     throw new FunctionalException("ProductsBloqueado", new string[] {String.Join(", ",
wUserList.ToArray()) });
  }
//(4)
wMarkProducts.FwkGuid = Guid.NewGuid();
wMarkProducts.TTL= Convert.ToInt32(ConfigurationManager.GetProperty("BlockingModel", "ProductionTTL"));
wMarkProducts.User = Environment.UserName;

//(5)
wEngine.Create(markProducts);
```

Código 1.0

1-Crea una marca y le asigna en el constructor el nombre de la tabla de NEGOSIO que se desea bloquear.

El primer parámetro es un identificador único para la marca, (GUID) en este caso se pasa **null** para que no asuma ningún valor ya que al momento de determinar si existe alguna marca por medio de los atributos de negocio por los que se bloqueo este valor no se lo conoce.

2- Se establecen los valores para verificar si existe una marca para:

Atributo: "ProductID"

Valor: 100

Tabla: "Productos"

3- Realiza la validación de si no hay otro usuario que haya tomado este producto con anterioridad a través de `FwkBlockingEngine.` Si existen usuario/s lanza una excepción funcional informando este caso.

4-Para el caso de que no esté bloqueada se crea finalmente la marca de bloqueo pero antes se le agregan:

Guid: identificador único.

UserName: Nombre de usuario que realiza la marca.

TTL: Tiempo de vida en este caso nos ayudamos del sistema de configuración del framework para obtener los TTL del grupo BlockingModel.

5-Se llama al engine para generar la marca.

Ahora en el momento de que el usuario decida realizar la actualización deberá realizar los siguientes pasos:

1-Verificar que no haya expirado su marca de bloqueo establecida en **código 1.0**. El GUID es el mismo que se genero en el paso (4) del **código 1.0** y se almaceno en la variable *pBlockingGuid* en algun momento cuando se realizo el bloqueo.

```
Guid wGuid = new Guid(pstrBlockingGuid);
FwkBlockingEngine wEngine = new FwkBlockingEngine();
if (!wEngine.Exists(wGuid, null))
    throw new FunctionalException("BlockingExpiro", String.Empty);
```

Código 1.1

2-Si no expiro entonces es posible realizar la actualización del producto y después de la actualización se deberá liberar la marca de bloqueo.

```
else
{
        ProductsDAC.Actualizar(pProductBE);
        wEngine.Remove(wGuid);
}
```

Código 1.2