MEMORIA ACTIVEMQ
PRÁCTICA

JAVIER MOLPECERES GOMEZ

48642626-H

Contenido

Introducción

En esta práctica se nos pide realizar un control de invernaderos en la cual utilice tecnología MOM. Se controlan 2 invernaderos los cuales tienen sensores de temperaturas y de humedad, envían mensajes a un control el cual comprobará si es necesario activar el ventilador o los deshumificadores.

Invernaderos

Los invernaderos controlan la humedad y la temperatura, en este caso se crean los datos según un timer que va modificando de manera aleatoria y sumando o restando temperatura y humedad según si están activados los ventiladores y los humificadores.

Podemos ver los distintos métodos que aumentan o reducen estos valores:

```
1 reference | jamolpe, 1 day ago | 1 author, 1 change
private void subirTemp()
{
    Random random = new Random();
    temp = temp + Math.Round((random.NextDouble() * (7 - 0) + 0), 2);
}

1 reference | jamolpe, 1 day ago | 1 author, 1 change
private void subirHumedad()
{
    Random random2 = new Random();
    humedad = humedad + Math.Round(random2.NextDouble() * (9 - 0) + 0);
}

1 reference | jamolpe, 12 hours ago | 1 author, 2 changes
private void bajarTemp()
{
    Random random = new Random();

    temp = temp - Math.Round((random.NextDouble() * (8 - 0) + 0), 2);

}

1 reference | jamolpe, 12 hours ago | 1 author, 2 changes
private void bajarHumedad()
{
    Random random2 = new Random();
     humedad = humedad - Math.Round(random2.NextDouble() * (9 - 0) + 0);
}
```

Estos invernaderos envían cada cierto tiempo (5 segundos) información sobre su temperatura y humedad, esperando una respuesta del control.

```
1 reference | jamolpe, 12 hours ago | 1 author, 7 changes
private void timer1_Tick(object sender, EventArgs e)
{

    if (!acventiladores)
    {
        subirTemp();
    }
    else
    {
        bajarTemp();
    }
    if (!achumificadores)
    {
        subirHumedad();
    }
    else
    {
        bajarHumedad();
    }

    lbbTemp.Text = temp.ToString() + " °C";
    lblHumedad.Text = humedad.ToString() + " %";


    ComprobarRespuestaHumificador();
    ComprobarRespuestaTemperaturas();

}
```

Podemos ver que en cada tick del reloj se llama a comprobar respuestas tanto de humificador como de temperaturas.

```
1 reference | jamolpe, 12 hours ago | 1 author, 1 change
public void ComprobarRespuestaTemperaturas()
{
    var mensaje = this.CreateTextMessage(txtNombreInver.Text + "|" + temp.ToString());
    var respuesta = EnviarMensajeTemp(mensaje);
    var res = respuesta as ITextMessage;

    if (res.Text == "true")
    {
        acventiladores = true;
        txtboxVentiladores.BackColor = Color.LimeGreen;
    }
    else
    {
        acventiladores = false;
        txtboxVentiladores.BackColor = Color.White;
    }

}
```

Para ello se llama a enviarMensajeTemp, método que envía la información de la temperatura(usando su correspondiente topic) y espera una respuesta de manera asíncrona.

```
public IMessage EnviarMensajeTemp(IMessage message, int timeout = 10000)
{
    //id mensaje
    var correlationID = Guid.NewGuid().ToString();
    message.NMSCorrelationID = correlationID;

    // a quien responder los mensajes
    message.NMSReplyTo = this.temporaryQueueTemp;

    // AsyncMessageHelper Ayuda al mapeo de los mensajes clase que he encotnrado por internet.
    using (var asyncMessageHelper = new AsyncMessageHelper())
    {
        // añade el async helper al bufer de respuestas.
        lock (this.responseBuffer)
            this.responseBuffer[correlationID] = asyncMessageHelper;

        // Enviar el mensaje al queue
        this.producerTemp.Send(message);

        // Esperamos a que nos llegue respuesta o acabe el tiempo de espera no problem al ser asincrono,10 segu
        asyncMessageHelper.Trigger.WaitOne(timeout, true);

        // Either the timeout has expired, or a message was received with the same correlation ID as the reques
        IMessage responseMessage;
        try
        {
            // The Message property on the async helper will not have been set if no message was received with
            if (asyncMessageHelper.Message == null)

            // The Message property on the async helper will not have been set if no message w
            if (asyncMessageHelper.Message == null)
                throw new TimeoutException("Timed out while waiting for a response.");

            // We got the response message, cool!
            responseMessage = asyncMessageHelper.Message;
        }
        finally
        {
            // Remove the async helper from the response buffer.
            lock (this.responseBuffer)
                this.responseBuffer.Remove(correlationID);
        }

        // Return the response message.
        return responseMessage;
    }
}
```

Por lo tanto, el invernadero podrá seguir trabajando sin ningún problema mientras espera la respuesta (como máximo durante 10 segundos), cuando reciba la respuesta se guardara en una cola de respuestas. Un dato importante es que se ha utilizado una clase para referenciar los mensajes, estos mensajes están formados por un identificador único y la propia respuesta, esto se utiliza para poder identificar cada mensaje con su petición y así no tener problemas de recibir respuestas de otro mensaje.

Por último, es importante mostrar cómo se han creado los topics que trabajan con el invernadero de manera que se comunican con el cliente.

```
1 reference | jamolpe, 15 hours ago | 1 author, 2 changes
public void ActiveMQInvernadero()
{

    String user = env("ACTIVEMQ_USER", "admin");
    String password = env("ACTIVEMQ_PASSWORD", "password");
    String host = env("ACTIVEMQ_HOST", "localhost");
    int port = Int32.Parse(env("ACTIVEMQ_PORT", "61616"));

    String destinationQueueTemp = "Temperatura";
    String destinationQueueHum = "Humedad";
    String destinationQueueConf = "Configuracion";

    this.responseBuffer = new Dictionary<string, AsyncMessageHelper>();

    String brokerUri = "activemq:tcp://" + host + ":" + port;
    var connectionFactory = new ConnectionFactory(brokerUri);

    try
    {
        connection = connectionFactory.CreateConnection();
        connection.Start();

        this.session = connection.CreateSession(AcknowledgementMode.AutoAcknowledge);
        var destinationTemp = session.GetDestination(destinationQueueTemp);
        var destinationHum = session.GetDestination(destinationQueueHum);
        var destinationConf = session.GetDestination(destinationQueueConf);
```

```
        this.producerTemp = session.CreateProducer(destinationTemp);
        this.producerTemp.DeliveryMode = MsgDeliveryMode.NonPersistent;
        this.producerHum = session.CreateProducer(destinationHum);
        this.producerHum.DeliveryMode = MsgDeliveryMode.NonPersistent;
        this.producerConf = session.CreateProducer(destinationConf);
        this.producerConf.DeliveryMode = MsgDeliveryMode.NonPersistent;

        this.temporaryQueueHum = session.CreateTemporaryQueue();
        this.temporaryQueueTemp = session.CreateTemporaryQueue();
        this.temporaryQueueConf = session.CreateTemporaryQueue();

        var responseConsumerTemp = session.CreateConsumer(temporaryQueueTemp);
        var responseConsumerHum = session.CreateConsumer(temporaryQueueHum);
        var responseConsumerConf = session.CreateConsumer(temporaryQueueConf);

        responseConsumerTemp.Listener += new MessageListener(responseConsumer_ListenerTemp);
        responseConsumerHum.Listener += new MessageListener(responseConsumer_ListenerHum);
        responseConsumerConf.Listener += new MessageListener(responseConsumer_ListenerConf);
    }
    catch(Exception ex)
    {

    }
}
```

Control

El control es el encargado de recibir la información de los invernaderos y devolver en caso necesario la respuesta de activación de ventiladores o humificadores. Para ello se han configurado diferentes topics para tratar tanto la configuración inicial que se le pasa al principio de cada invernadero como la humedad y la temperatura.

```
1 reference | jamolpe, 52 minutes ago | 1 author, 3 changes
public void IniciarActiveMQServer()
{

    Console.WriteLine("Iniciando control..");

    String destinationQueueTemp = "Temperatura";
    String destinationQueueHumedad = "Humedad";
    String destinationQueueConfiguracion = "Configuracion";

    String user = env("ACTIVEMQ_USER", "admin");
    String password = env("ACTIVEMQ_PASSWORD", "password");
    String host = env("ACTIVEMQ_HOST", "localhost");
    int port = Int32.Parse(env("ACTIVEMQ_PORT", "61616"));

    String brokerUri = "activemq:tcp://" + host + ":" + port + "?transport.useLogging=true";

    var connectionFactory = new NMSConnectionFactory(brokerUri);
    IConnection connection;

    try
    {
        connection = connectionFactory.CreateConnection();
        connection.Start();

        this.session = connection.CreateSession(AcknowledgementMode.AutoAcknowledge);

        var queuetemp = session.GetDestination(destinationQueueTemp);
        var queuehum = session.GetDestination(destinationQueueHumedad);
        var queueconf = session.GetDestination(destinationQueueConfiguracion);

        this.replyProducer = this.session.CreateProducer();
        this.replyProducer.DeliveryMode = MsgDeliveryMode.NonPersistent;

        var consumertemp = this.session.CreateConsumer(queuetemp);
        var consumerhum = this.session.CreateConsumer(queuehum);
        var consumerconf = this.session.CreateConsumer(queueconf);

        consumertemp.Listener += new MessageListener(Calculo_Temp);
        consumerhum.Listener += new MessageListener(Calculo_Hum);
        consumerconf.Listener += new MessageListener(Calculo_Config);
    }
    catch (Exception ex)
    {
        Console.WriteLine("error " + ex.ToString());
    }
}
```

Podemos ver que según el tipo de topic tiene un listener asociado que realizara un trabajo.

```
1 reference | jamolpe, 55 minutes ago | 1 author, 3 changes
public void Calculo_Hum(IMessage message)
{
    try
    {

        var response = this.session.CreateTextMessage();
        var textMessage = message as ITextMessage;
        string[] mns = textMessage.Text.Split('|');
        String info = "Humedad invernadero " + mns[0] + " es -> " + mns[1];


        if (textMessage == null)
            response.Text = "false";
        else
        {
            int hum = int.Parse(mns[1]);
            if (hum > referenciasHume[mns[0]])
            {
                response.Text = "true";
                info += "\n";
                info += "Activando deshumificador";
            }
            else
            {
                response.Text = "false";
            }
        }

        response.NMSCorrelationID = message.NMSCorrelationID;

        this.replyProducer.Send(message.NMSReplyTo, response);
        rellenarinfo(info);
    }
    catch (NMSException ex)
    {
        Console.WriteLine("error " + ex.ToString());
    }
}
1 reference | jamolpe, 56 minutes ago | 1 author, 3 changes
```

Un listener a resaltar es el de configuración, el cual guarda en 2 diccionarios la información sobre la temperatura y la humedad máxima de cada invernadero, estos diccionarios se utilizarán para comprobar si se han superado estas temperaturas o humedades.

```
1 reference | jamolpe, 58 minutes ago | 1 author, 5 changes
public void Calculo_Config(IMessage message)
{
    try
    {
        String info = "";

        var response = this.session.CreateTextMessage();


        var textMessage = message as ITextMessage;

        if (textMessage == null)
            response.Text = "false";
        else
        {
            string[] datos = textMessage.Text.Split('|');
            info = "Configuracion realizada para el invernadero " + datos[0]+"\n";
            referenciasTemp.Add(datos[0],Int32.Parse(datos[1]));
            referenciasHume.Add(datos[0], Int32.Parse(datos[2]));
            info += "Humedad maxima = " + datos[2]+"\n";
            info += "Temperatura maxima = " + datos[1] + "\n";
        }


        response.NMSCorrelationID = message.NMSCorrelationID;

        this.replyProducer.Send(message.NMSReplyTo, response);
        rellenarinfo(info);

            rellenarinfo(info);
        }
        catch (NMSException ex)
        {
            Console.WriteLine("error " + ex.ToString());
        }
    }
```

Despliegue

Para realizar el despliegue es necesario activar activeMQ luego el cliente y después tantos invernaderos como queramos. Se activaran de color verde si están activados los respectivos ventiladores o humificadores.