ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
CENTRO DE INVESTIGACIÓN CIENTÍFICA Y TECNOLÓGICA

Sistema Para Control De Tareas O Proyectos Y Medición De Rendimiento Basado En Resultados

William George Murillo Párraga
Facultad de Ingeniería en Electricidad y Computación
Escuela Superior Politécnica del Litoral
Guayaquil, Ecuador
wmurillo@espol.edu.ec

Carmen Karina Vaca Ruiz
Facultad de Ingeniería en Electricidad y Computación
Escuela Superior Politécnica del Litoral
Guayaquil, Ecuador
cvaca@espol.edu.ec

Resumen

El propósito de este proyecto de tesis de grado es presentar una herramienta para administrar tareas dentro de un ambiente laboral, sirviendo como alternativa de comunicación y control entre líderes de grupo y sus colaboradores. Para el desarrollo del proyecto y la definición de ideas que lo hicieran realidad, se tomó como base conceptos de control, delegación, comunicación y medición. Si bien el mercado ofrece algunas herramientas para cubrir esta necesidad, se busca ofrecer algo más sencillo, pero que a su vez no deje de ser práctico y útil.

Palabras Claves: *ASP.NET, Software, Tareas, Web, Actividades*.

Abstract

The purpose of this Project of thesis is to offer a tool to administrate tasks into a business environment, working as an alternative for communication and control between leaders and collaborators into a workgroup. To develop this project and to define the ideas to make this true, concepts of control, delegation, communication and measurement were taken as fundamentals. Actually the market offers some tools satisfy the necessity, but this tool intends to be simple but useful and practical.

1. Introducción

En toda organización existen siempre tareas por cumplir, desde algunas críticas para el desenvolvimiento del negocio hasta otras de menor importancia, pero que son parte de las obligaciones del empleado y cuentan al momento de evaluar el rendimiento enfocado en resultados obtenidos.
Este proyecto se enfoca en ayudar tanto a empleados a auto controlarse y organizarse así como a líderes de grupo a controlar y medir el desempeño de sus colaboradores.

Durante la investigación de herramientas que ayuden a controlar el cumplimiento de tareas, el patrón que se detectó es que por querer ser muy completas se vuelven muy complejas, llenando a los usuarios de una variedad increíble de opciones. Tal como ocurre en herramientas de uso diario como Microsoft Word o Excel, solo las opciones más básicas son las usadas con frecuencia. Otras en cambio, buscando ser muy prácticas, se vuelven muy básicas.

En este artículo se explica cómo se llevó a cabo la implementación de una herramienta para controlar tareas, que busca ser práctica, pero a la vez útil. Se explicarán los conceptos que fundamentan la idea y detalles técnicos de su implementación.

2. Contenido

2.1. Bases para crear el proyecto

Las bases en las cuales se fundamenta este proyecto son delegación, comunicación, control y medición. Partimos de que el control es un elemento esencial en todo tipo de actividad. Esto lo complementamos considerando que no hay control adecuado si no existe una proporcionada delegación, acompañada por una comunicación que aunque no sea fluida en un 100%, se mantenga activa. Finalmente, aquello que se puede medir, se puede manejar y por tanto se puede mejorar.

2.2 Alcance del proyecto

El proyecto contempla una herramienta en la cual se puedan crear múltiples grupos de trabajo, a los cuales se les designa un líder para llevar un control de actividades de ese grupo. Los líderes pueden asignar tareas a los colaboradores de su grupo y luego se lleva el control a través del sistema.

2.2.1 Grupos de trabajo. Se pueden crear múltiples grupos de trabajo. Una persona puede pertenecer a más de un grupo de trabajo, pudiendo ser solo colaborador o también líder del grupo o líder alterno para cubrir al líder en ciertas funciones durante su ausencia.

2.2.2 Control de tareas. El cumplimiento de las tareas se controla a través de estados. Cuando se crea una tarea está en estado Nuevo. Cuando se le asigna la primera fecha de compromiso tiene el estado de Abierto. Cuando tiene más de una fecha de compromiso pasa al estado de Retrasado y al completarla se asigna el estado de Cerrado. El seguimiento de la tarea se realiza a través de comentarios que se solicitan de manera semanal.

2.2.3 Medición de resultados. Se incluyen 2 métricas para controlar de manera macro el avance de las tareas. Una métrica permite conocer cuantas tareas de las asignadas realmente se han cumplido en un rango de tiempo determinado. La otra métrica permite detectar las personas que más se retrasan con sus tareas.

2.3. Descripción de la arquitectura 3 capas de ASP.NET

ASP.NET [1] es una plataforma para desarrollo Web, una de las más poderosas del mercado, que provee todas las herramientas y servicios necesarios para construir aplicaciones Web empresariales, comerciales o de tipo personal. ASP.NET se nutre y beneficia de todas las características que ofrece el Framework .Net. Para el desarrollo de este proyecto se usó el modelo de 3 capas [2] presentado en el portal oficial de ASP.NET, el cual se compone de:

- Capa de Acceso a Datos: Se compone por archivos de tipo Dataset, los cuales en esencia son esquemas XML que representan un medio de acceso a la base de datos.
- Capa Lógica: Se maneja con clases. Esta capa invoca a la capa de acceso a datos y es la que realiza el procesamiento debido de la información para enviarla a la capa de presentación.
- Capa de Presentación: se compone de formularios web, los cuales se componen de dos archivos, uno el archivo .aspx que contiene todo el XHTML que genera la información y el otro archivo es el .aspx.vb que contiene el procesamiento que se realiza para la interacción del usuario con la aplicación.

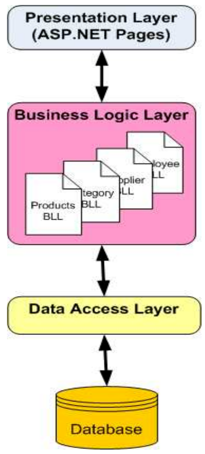

Figura 1. Estructura del Modelo 3 capas de ASP.NET.

2.3.1. Justificación del uso de la arquitectura 3 capas de ASP.NET. Existe una gran cantidad de ventajas que se obtienen de usar ASP.NET para los desarrollos, pero las que justifican el uso de esta tecnología para el proyecto actual son [3]:

- Compatibilidad con herramientas de primer nivel: El marco de trabajo de ASP.NET se complementa con un diseñador y una caja de herramientas muy completos en el entorno integrado de programación (IDE) de Visual Studio.
- Eficacia y flexibilidad: Debido a que ASP.NET se basa en Common Language Runtime, la eficacia y la flexibilidad de toda esa plataforma se encuentra disponible para los programadores de aplicaciones Web.
- Simplicidad: ASP.NET facilita la realización de tareas comunes, desde el sencillo envío de formularios y la autenticación del cliente hasta la implementación y la configuración de sitios.
- Seguridad.- Junto con ASP.NET viene incluido un módulo de seguridad que facilita y ahorra gran cantidad de tiempo y esfuerzo en temas relacionados al manejo de la seguridad y validación de usuarios.
- Menos Líneas de código: Con la tareas comunes que se encapsulan en controles y que pueden ser usados a través de todo el sitio Web, muchas de las cosas que se hacen tradicionalmente ahora se realizan con mucho menos código que en versiones anteriores de ASP u otras herramientas.
- Rendimiento y Escalabilidad.- ASP.NET se ha creado para desempeñarse usando un modelo de ejecución compilado para manejar solicitudes de las páginas. Es compatible con procesadores y servidores de 64-bits. Incluye además mejoras en el manejo de la caché.

2.4. Modelo de acceso a datos

Para el acceso a datos, comúnmente en las aplicaciones se crea una clase que instancia una conexión a la base de datos y otras clases en las cuales se generan manualmente las sentencias SQL. Luego se realizan procedimientos para pasar los parámetros respectivos, abrir la conexión a la base de datos, ejecutar la sentencia, capturar los datos si es que aplica y luego volver a cerrar la conexión. A esto se le agrega el control que por lo general se realiza para capturar excepciones. Realizar este tipo de procedimientos toma en los casos más pequeños unas 12 líneas de código. En aplicaciones que usan sentencias SQL más complejas se puede tomar de 30 a 50 líneas de código y esto por cada acción que se desee realizar.

Para ahorrar tiempo y líneas de código, en este proyecto se aprovecha la ventaja que provee ASP.NET a través de los esquemas XML que representan un Dataset [4].

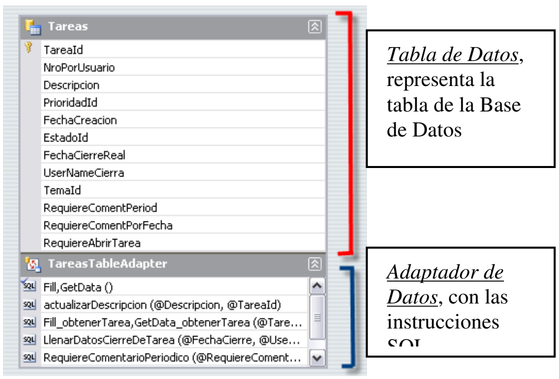

Figura 2. Representación de un Adaptador de Datos en un Dataset.

El Dataset físicamente es un archivo en el cual se crean uno o más adaptadores, los cual generan una representación de la tabla de la base de datos en la aplicación. La representación de la tabla de datos es la que se marca con rojo en la Figura 2.

Luego en el adaptador se pueden agregar múltiples sentencias SQL, que se muestran en la parte de color azul de la figura 2. Para crear las sentencias SQL se usa una opción que brinda Visual Studio la cual permite generar todo tipo de sentencias SQL y probarlas antes de grabarlas. Esto nos ahorra gran cantidad de tiempo y líneas de código. A cada sentencia SQL se la crea como un método y los parámetros se pasan como parámetros de cualquier método. Con esto nos ahorramos instrucciones para instanciar parámetros, abrir y cerrar la conexión, crear la sentencia SQL, etc., lo que generalmente se implementaría como se muestra en la Figura 3.

```
Public Function obtenerTarea(ByVal TareaId As Int32) As TareasDataTable
    Dim dc As New SqlCommand
    Dim cn As SqlConnection
    Dim dr As SqlDataReader
    Dim dt As New TareasDataTable

    cn = New SqlConnection(ConfigurationManager.ConnectionStrings("TareasCN").Connect:
    dc.CommandType = CommandType.Text
    dc.Connection = cn
    dc.CommandText = "SELECT t.[TareaId], t.[NroPorUsuario], t.[Descripcion], "
    dc.CommandText &= "t.[PrioridadId], t.[FechaCreacion], t.[EstadoId],"
    dc.CommandText &= "t.[FechaCierreReal], t.[UserNameCierra], t.[TemaId],"
    dc.CommandText &= "t.[RequiereComentPeriod], t.[RequiereComentPorFecha], "
    dc.CommandText &= "t.[RequiereAbrirTarea]"
    dc.CommandText &= "FROM [ta_Tareas] t"
    dc.CommandText &= "WHERE t.Activo = 1     AND  t.TareaId = @TareaId"

    dc.Parameters.Add(New SqlParameter("@TareaId", TareaId))

    dc.Connection.Open()
    dr = dc.ExecuteReader()
    If dr.HasRows Then
        'instrucciones para llenar el TareasDataTable
    End If
    dc.Connection.Close()
    Return dt

End Function
```

Figura 3. Código para una llamada a la Base de Datos.

2.5. Procesamiento de la información en la capa lógica

Como es común en una capa lógica de trabajo, aquí se procesa toda la información que proviene de la base de datos a través de la capa de acceso a datos.
La comunicación con la capa de acceso a datos se realiza creando instancias por cada adaptador según se requiera [5]. Esto se consigue definiendo una propiedad de solo lectura del tipo del adaptador al que hacemos referencia. La Figura 4 muestra cómo se instancia el adaptador TareasTableAdapter mostrado en la Figura 2

```
Protected ReadOnly Property Tareas() As TareasTableAdapter
    Get
        If _TareasAdapter Is Nothing Then
            _TareasAdapter = New TareasTableAdapter()
        End If
        Return _TareasAdapter
    End Get
End Property
```

Figura 4. Código para hacer instanciar un adaptador de datos.

Para establecer una comunicación adecuada con la capa de presentación, en cada método implementado se identifique claramente su tipo, es decir si es un método que retorna datos, que actualiza, inserta o elimina información. Esto se lo logra ubicando sobre cada método una instrucción como la que se muestra en la Figura 5.

```
<System.ComponentModel.DataObjectMethodAttribute _
(System.ComponentModel.DataObjectMethodType.Select, True)> _
```

Figura 5. Instrucción para identificar tipo de método en una clase.

En esta instrucción se debe especificar el tipo de función, los valores posibles son:

- Select: Para funciones que retornan registros
- Insert: Para funciones que graban registros
- Update: Para funciones que actualizan registros
- Delete: Para funciones que eliminan registros

El último parámetro es un valor de verdadero y falso, con el cual se indica si el método es el método principal de la clase según su tipo.
Para completar la definición de las funciones, se necesita agregar al momento de declarar la clase una instrucción que indique que la clase provee datos. Estos datos serán luego utilizados por la capa de presentación. La Figura 6 muestra la sentencia que se debe especificar antes de declarar una clase.

```
<System.ComponentModel.DataObject()> _
Public Class Tarea
```

Figura 6. Código para conectar una clase con la capa de presentación.

2.6. Capa de presentación: Formularios Web.

Para presentar a los usuarios la información que viene a través de la capa lógica usamos unos componentes conocidos como ObjectDatasource.
Un ObjectDatasource puede tener referenciados un método para retornar información, uno para actualizar, uno para eliminar y uno para crear. Se le debe especificar la clase de la capa lógica en la cual están los métodos. La Figura 7 muestra las propiedades que tiene un ObjectDatasource.

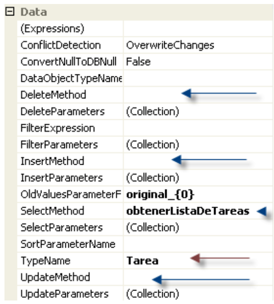

Figura 7. Propiedades de un ObjectDatasource.

En la Figura 7 se muestra con línea azul el lugar donde se especifican los métodos según el tipo y con línea roja donde se especifica la clase que contiene a

los métodos. Toda esta información es asignada automáticamente por el Visual Studio al momento de crear el ObjetcDatasource. Esto nos permite ahorrarnos muchas líneas de código al momento de presentar la información al usuario.
Un ObjectDatasource se puede usar como fuente de datos para cualquiera de los controles de ASP.NET

2.6.1 Ajax en la capa de presentación. Aplicar AJAX con ASP.NET es una tarea muy sencilla. Uno mantiene la programación normal que realizaría si no usa AJAX [6]. Para que se genere contenido asincrónico lo que se debe hacer es primero agregar en la página un administrador de script, usando la instrucción que se muestra en la Figura 8.

```
<asp:ScriptManager ID="ScriptManager1" runat="server" />
```

Figura 8. Código que instancia administrador de scripts ajax.

Luego la sección de la página que deseamos manejar de manera asincrónica la ubicamos dentro de la etiqueta UpdatePanel y su sub-etiqueta ContentTemplate, tal como se muestra en la Figura 9.

```
<asp:UpdatePanel ID="UpdatePanel2" runat="server">
    <ContentTemplate>
        <!-- CONTENIDO QUE TRABAJARÁ ASINCRÓNICAMENTE CON AJAX-->
    </ContentTemplate>
</asp:UpdatePanel>
```

Figura 9. Etiquetas para marcar zona de procesamiento asincrónico.

De esta forma, el Framework de ASP.NET se encarga de administrar la acción de manera asincrónica sin que sea necesario que el desarrollador cree funciones en Java Script como se requeriría comúnmente en otra plataforma

3. Conclusiones

Gracias a ASP.NET ha sido posible ahorrar más del 50% del tiempo que hubiera tomado desarrollar esta herramienta usando otras tecnologías como Java o PHP. El IDE Visual Studio 2005 y sus múltiples controles de usuario simplifican muchas tareas, permitiendo a los desarrolladores enfocarse en generar ideas para solucionar temas de negocio y no temas de implementación.
Aplicar conceptos de arquitectura para el desarrollo resultó beneficioso al momento de hacer las modificaciones que resultaron de las observaciones hechas por los usuarios durante las pruebas del sistema.

.

4. Referencias

[1] Guía de ASP.NET, Novedades en ASP.NET 2.0, http://quickstarts.asp.net/QuickStartv20/aspnet/doc/whatsnew.aspx
[2] Tutorial de Acceso a Datos con ASP.NET, http://www.asp.net/learn/data-access/
[3] Cambios en la Función de ASP.NET 2.0, http://msdn2.microsoft.com/es-es/library/aa479401(en-us).aspx
[4] Crear una Capa de Acceso a Datos con ASP.NET, http://www.asp.net/learn/data-access/tutorial-01-vb.aspx
[5] Crear una Capa Lógica de Negocios con ASP.NET, http://www.asp.net/learn/data-access/tutorial-02-vb.aspx
[6] ASP.NET AJAX, http://www.asp.net/ajax/