

COMPARTIMOSS

Revista especializada en Tecnologías SharePoint



ENTREVISTA

NOVEDADES EN SQL SERVER 2014

NUEVAS API'S PARA DESARROLLADORES EN OFFICE 365 PARTE I

EXPONIENDO DATOS BINARIOS CON BCS (II)

STAFF

CompartiMOSS es una publicación independiente de distribución libre en forma electrónica. Las opiniones aquí expresadas son de estricto orden personal, cada autor es completamente responsable de su propio contenido.

DIRECCIÓN GENEAL

- Gustavo Velez
- Juan Carlos Gonzalez
- Fabian Imaz
- Alberto Diaz

Contacte con nosotros

revista@compartimoss.com
gustavo@gavd.net
jcgonzalezmartin1978@hotmail.com
fabian@siderys.com.uy
adiazcan@hotmail.com

BLOGS

<http://www.gavd.net>
<http://geeks.ms/blogs/jcgonzalez>
<http://blog.siderys.com>
<http://geeks.ms/blogs/adiazmartin>

REDES SOCIALES

Facebook:
<http://www.facebook.com/group.php?gid=128911147140492>
LinkedIn:
<http://www.linkedin.com/groups/CompartiMOSS-3776291>
Twitter:
@CompartiMOSScom

Contenidos

03

Editorial

Cumpliendo con nuestro propósito de innovación y evolución constante para CompartiMOSS

04. Nuevas API's para desarrolladores de Office 365 Parte I

10. Cómo consumir los servicios REST de SharePoint Online 2013 desde un Workflow en SharePoint Designer

16. Ajuste de la Relevancia en el Motor de Búsqueda de Sharepoint 2013

22

SharePoint 2013 sobre Azure IAAS

24. SharePoint 2013: ¿Solución o Aplicación?

27. Entrevista a Andrés Rojas

29. SharePoint 2010 Deferred site collection upgrade with custom SharePoint solutions

30

Novedades en SQL Server 2014

32. Novedades en el manejo de Bibliotecas de Documentos en SharePoint Online

34. Buenas Prácticas en Infraestructura en SharePoint 2013 (Parte 2)

44. Exponiendo datos binarios con BCS (II)

52. El autor destacado del Número 19



03

Editorial

Cumpliendo con nuestro propósito de innovación y evolución constante para CompartiMOSS, este número de la revista estamos comenzando a abrir las puertas a artículos sobre SharePoint en otros idiomas. Después de numerosas discusiones entre las personas que dirigimos la revista, sopesar los pros y los contras y evaluar las diferentes posibilidades, hemos decidido que la evolución de la revista no solo nos permite dar la bienvenida a opiniones y experiencias escritas en un idioma diferente al castellano, sino que también nuestros lectores merecen obtener información valiosa de autores en un idioma diferente.

Desde el principio CompartiMOSS ha tenido un carácter internacional, no solamente porque nuestros lectores vienen de numerosos países en dos continentes, sino también porque los que manejamos la revista tienen tres nacionalidades diferentes y vivimos y trabajamos en países totalmente diferentes. También se puede ver como un ejemplo de colaboración, de la cual SharePoint es no solamente el centro físico de la información, sino la herramienta por excelencia para poder trabajar de esta forma. La expansión hacia artículos en otros idiomas diferentes al castellano no significa que nuestro lema principal, la distribución de información sobre SharePoint en español, deja de ser válido, sino que significa un enriquecimiento de la calidad y cantidad de la información que presentamos. En un principio no vamos a publicar más de uno (máximo dos) artículos en un idioma diferente al nuestro en cada número, y nos limitaremos a artículos en inglés y las lenguas más cercanas al castellano, tanto geográfica, como lingüísticamente hablando.

Esperamos que, como de costumbre, se diviertan tanto leyendo este número de CompartiMOSS como nosotros lo hacemos creándola.

EL EQUIPO EDITORIAL DE COMPARTIMOSS





04

Nuevas API's para desarrolladores en Office 365 Parte I

Resumen

Este es el primero de una serie de artículos que tienen como objetivo introducir las nuevas APIs que van a estar disponibles en Office 365. Estas nuevas APIs facilitan el acceso a elementos de Exchange (calendarios, e-mails, contactos), a ficheros de SharePoint y al Directorio Activo de Azure de una forma relativamente simple y muy sencilla. Uno de los beneficios de estas APIs es que se pueden utilizar desde fuera del contexto de Office 365, lo que permite que se puedan desarrollar Apps en diversas tecnologías y plataformas como Windows 8.1, Android o iOS y aprovechar en ellas todas las ventajas de Office 365. En este primer artículo vamos a desglosar todas las opciones disponibles en las API REST de Exchange Online.

Artículo

Nota: Estas APIs están actualmente en fase de Preview y pueden sufrir modificaciones cuando se lance la versión definitiva. Naturalmente, no se recomienda el uso de ninguna de estas APIs en un entorno de producción.

Introducción

En la Conferencia de SharePoint que tuvo lugar en Las Vegas el pasado mes de Abril, Microsoft presentó muchas novedades para nuestro servidor favorito sobre todo en la versión Online: nuevos formularios para sustituir a InfoPath, Office Graph, novedades en las Apps, versión de SDK para Android, mejoras en la API REST, novedades en NAPA y la aparición de nuevas APIs para Office 365.

Respecto a las nuevas APIs, nos encontraremos con las siguientes:

- Exchange Online: Permitir el acceso, añadir y modificar elementos tanto a nivel de Calendarios y de Contactos como enviar y consultar correos electrónicos.
- SharePoint Online: Facilitar el acceso a carpeta y ficheros. Representa un rediseño del almacenamiento y la administración de archivos a través de la API de SharePoint.
- Active Directory: Consulta, alta y modificación de usuarios en el Directorio Activo de Azure.

Además de estas APIs, el uso de las mismas es posible gracias a otra de las novedades que se han introducido como es la posibilidad de implementar OAuth fuera del contexto de SharePoint Online. Es decir, tengo una aplicación fuera del contexto de Office 365, y le puedo dar permisos para utilizar

estos datos. Esta técnica de delegación de las credenciales está presente en casi todas las aplicaciones móviles, como por ejemplo cualquier App de Twitter, Instagram, etc. Cómo se implementa esta delegación de credenciales no es objeto de este artículo, para más información de cómo realizarlo se puede consultar este enlace: <http://msdn.microsoft.com/es-ES/library/office/dn605895>

Si damos tanta importancia a estas APIs es porque, desde que se presentó el nuevo modelo de Apps y la Office Store, Microsoft apenas había presentado novedades. Algunas de estas APIs (como por ejemplo la API de Exchange Online) eran continuas reclamaciones que le hacíamos los desarrolladores para poder implementar aplicaciones interesantes y de calidad en la Office Store utilizando Exchange Online. Además el hecho de que estas APIs puedan ser Cross-Plataforma da un motivo más a los clientes para unirse a Office 365, bien en un entorno híbrido, bien en un entorno nube.

Desgranando las API de Exchange Online

La API que se incluyen dentro de Exchange es la que más novedades incorpora y, quizás, de la que más partido se le pueda obtener. Principalmente porque Exchange fue de los primeros productos que se decidió llevar a la nube, con lo cual, la base instalada es bastante amplia.

Mail REST API

La Mail REST API proporciona acceso total a los mensajes de correo electrónico en el buzón de un usuario en Exchange Online. Tenemos los siguientes Endpoints disponibles:

• Obtener el e-mail:

Podemos obtener el correo electrónico de dos maneras diferentes. Se puede solicitar un conjunto de mensajes de correo electrónico desde una carpeta, opcionalmente filtro o una ordenación, o puede solicitar una dirección de correo electrónico específica por su propiedad ID.

1. Obtener los mails de la carpeta Inbox:

```
GET https://outlook.office365.com/ews/odata/Me/Inbox/Messages
Accept: application/json;odata.metadata=full
```


2. Obtener un e-mail en particular, para ello tendremos que tener el Identificador del e-mail que queremos consultar:

```
GET https://outlook.office365.com/EWS/odata/Me/Messages('ID del mensaje')
Accept: application/json;odata.metadata=full
```

Otra de las posibilidades de que se dispone es, dado que hace uso del estándar OData versión 4.0, es la de poder realizar filtros, paginación, ordenación y número de elementos a mostrar dentro de la propia petición como por ejemplo:

```
GET https://outlook.office365.com/ews/odata/Me/Inbox/Messages?$orderby=DateTimeReceived&$filter=Importance eq Microsoft.Exchange.Services.OData.Model.Importance'Low'&$select=Subject,Importance,DateTimeReceived&$top=5
Accept: application/json;odata.metadata=full
```

En el ejemplo propuesto ordenamos los mensajes por fecha de creación haciendo uso de \$orderby, filtramos mediante el parámetro \$filter por los elementos que son de importancia "Low" y mostramos los siguientes campos: Subject, Importance y DateTimeReceived utilizando \$select. Indicamos el número de elementos que nos traemos con la sentencia \$top donde en este caso nos trae los primeros 5 elementos.

Nota: El número de elementos que nos podemos traer por petición es de un máximo de 500. Por defecto está limitado a 50, pero se puede incrementar hasta 500.

- **Crear un e-mail:**

Para crear un mail tendremos que crear un objeto JSON con la estructura de un e-mail: Mensaje, Cuerpo y Recipientes. El objeto tendría la siguiente estructura:

```
{
  "@odata.type": "#Microsoft.Exchange.Services.OData.Model.Message",
  "Subject": "Articulo Compartimos para utilizar las nuevas APIS de Office 365",
  "Importance": "High",
  "Body": {
    "ContentType": "HTML",
    "Content": "Aquí está el articulo leerlo y cualquier cosa me decís"
  },
  "ToRecipients": [
    {
      "Name": "Alberto Diaz",
      "Address": "adiazcan@compartimoss.com"
    },
    {
      "Name": "JC Gonzalez",
```

```
    "Address": "jcg1981@compartimoss.com"
  },
  "Name": "Fabian Imaz",
  "Address": "Fabian.imaz@compartimoss.com"
},
{
  "Name": "Gustavo Velez",
  "Address": "Gustavo.velez@compartimoss.com"
}
]
}
```

Sobre este objeto realizaremos la siguiente petición:

```
POST https://outlook.office365.com/ews/odata/Me/Drafts/Messages
Accept: application/json;odata.metadata=full
Content-Type: application/json;odata.metadata=full
```

NOTA: En el momento que creas un e-mail por defecto se crea dentro de la carpeta de borradores (se puede modificar para que directamente se envíe el mail directamente estableciendo el parámetro "MessageDisposition" al valor "SendAndSaveCopy", el valor por defecto es "SaveOnly").

- **Actualizar un e-mail:**

Una vez tenemos un mensaje en la carpeta de borradores, podemos realizar cualquier modificación del mismo, por un lado, generamos el objeto JSON con los campos a modificar, por ejemplo, vamos a modificar el Cuerpo del mensaje:

```
{
  "Body": {
    "ContentType": "HTML",
    "Content": "Vamos a modificar el mensaje!!"
  }
}
```

Con el objeto creado se realiza la siguiente petición:

- **Eliminar un e-mail:**

Se puede eliminar correo electrónico, simplemente enviando una solicitud DELETE para la dirección de la entidad de mensajes como por ejemplo, la siguiente petición.

```
DELETE https://outlook.office365.com/ews/odata/Me/Messages('Identificador del Mensaje')
```

- **Enviar un e-mail:**

Mediante la API se pueden enviar e-mails de dos formas:

- 1. Seleccionando el Identificador del e-mail :**

```
POST https://outlook.office365.com/ews/odata/Me/Messages('IdentificadordelMensaje')
```

2. Indicando el parámetro MessageDisposition=SendAndSaveCopy dentro de la petición de Crear un e-mail:

```
POST https://outlook.office365.com/ews/odata/Me/Drafts/Messages? MessageDisposition=SendAndSaveCopy
Accept: application/json;odata.metadata=full
Content-Type: application/json;odata.metadata=full
```

Calendar REST API

Con el Calendar API vamos a tener acceso tanto a los grupos del Calendario como los propios elementos, desde la propia API se va a poder contestar a Reuniones o Meetings propuestos.

• Obtener los grupos de Calendario:

En Outlook y Outlook Web App, los usuarios pueden añadir varios calendarios en un solo grupo de calendario. Esto hace que sea más fácil para los usuarios ver rápidamente todos los calendarios dentro del grupo. Se puede acceder a los grupos de calendarios en un buzón de correo a través de la propiedad CalendarGroups en el buzón. El siguiente ejemplo solicita todos los grupos de calendario en el buzón del usuario autenticado.

```
GET https://outlook.office365.com/ews/odata/Me/CalendarGroups
Accept: application/json
```

• Obtener los Calendarios:

Se puede acceder a los calendarios a través de la propiedad "Calendar", ya sea en el buzón o en un grupo de calendarios específicos. También puede acceder al calendario predeterminado del usuario a través de la propiedad Calendar en el buzón. El siguiente ejemplo solicita a todos los calendarios en el buzón.

```
GET https://outlook.office365.com/ews/odata/Me/Calendars
Accept: application/json
```

• Obtener los elementos de un calendario:

Con el Calendar REST API se permite crear, actualizar, eliminar, mover y copiar los elementos del calendario. También puede aceptar, rechazar o aceptar provisionalmente las solicitudes de reunión. Los elementos del calendario se dividen en las siguientes categorías:

- Citas - Una entrada básica en un calendario que no tiene asistentes. Se puede acceder a las citas a través de la propiedad Events en un calendario o el buzón.

- Reuniones - una cita con los asistentes. Las convocatorias de reunión se envían cuando se crea una reunión, y los asistentes tendrán la posibilidad de aceptar, rechazar o aceptar provisionalmente la solicitud. Se puede acceder a reuniones a través de la propiedad eventos en un calendario o el buzón.

- Reunión solicitudes - un elemento de calendario que se envía a los asistentes cuando el organizador crea, actualiza o cancela una reunión. Se puede acceder a las solicitudes de reunión a través de la propiedad de los mensajes en las carpetas o el buzón.

A continuación vamos a ver algunas de las operaciones disponibles en la API:

Consultar los elementos del calendario: Mediante la propiedad de Events obtenemos los elementos que tenemos en nuestros calendarios. Además, de la misma forma que con el mail, al hacer uso de OData es posible utilizar la propiedad "filter" para filtrar los elementos que se devolverán en la respuesta.

```
GET https://outlook.office365.com/ews/odata/Me/Calendar/Events?$filter=Start
le 2014-02-14T05:00:00Z and End ge
2014-02-10T05:00:00Z
Accept: application/json
```

Actualizar los elementos: Para actualizar un elemento de calendario, hay que realizar una petición PATCH para la URL del evento. En esta petición hay que incluir un objeto JSON con las propiedades para actualizar. Si actualiza una reunión en la que el usuario es el organizador, se envía una actualización a todos los asistentes. El ejemplo siguiente actualiza la ubicación de una reunión.

```
PATCH https://outlook.office365.com/ews/odata/Me/Calendar/Events('AAMkAGI3...')
Accept: application/json;odata.metadata=full
Content-Type: application/json;odata.metadata=full
Objeto JSON para actualizar la ubicación de la reunión
{
  "Location": {
    "DisplayName": "Encamina Canarias"
  }
}
```

Responder a una petición de reunión: Para responder a una petición hay que realizar varios pasos debido a que una Petición de reunión es un Objeto Mensaje y no un objeto Calendario. Por lo que en primer lugar, tendremos que consultar el mensaje. En dicho mensaje tendremos, por un lado en la propiedad "MeetingMessageType" y, en

la siguiente propiedad, tendremos un EventID que es el Identificador de la solicitud (posteriormente lo utilizaremos para contestar a dicha solicitud).

Una vez hemos consultado el mensaje y tenemos la información relativa al EVENTID podemos enviar una contestación a dicha convocatoria. Como por ejemplo:

```
POST https://outlook.office365.com/ews/odata/Me/Events('Identificador')/Accept?MessageDisposition=SendAndSaveCopy
Accept: application/json
Content-Type: application/json;odata.metadata=full
Expect: 100-continue
```

```
{
  "Comment": "Por mi parte perfecto. Allí estaré!"
}
```

Contact REST API

Con la Contact REST API puedes consultar, añadir, modificar y eliminar cualquier contacto.

• Consultar los Contactos:

Para obtener contactos, enviar una petición GET a la propiedad contactos de la carpeta de contactos. Dentro de la petición se puede utilizar el parámetro "Select" para indicar las propiedades que queremos que se devuelvan en la respuesta.

```
GET https://outlook.office365.com/ews/odata/Me/Contacts?$select=DisplayName,EmailAddress1,Birthday,Categories
Accept: application/json
```

• Crear un Contacto:

Para añadir un contacto, hay que enviar una solicitud POST en la que se envía un objeto JSON indicando que ese objeto es de tipo Contacto más la representación del contacto.

```
POST https://outlook.office365.com/ews/odata/Me/Contacts
Accept: application/json
Content-Type: application/json;odata.metadata=full

{
  "@odata.type": "#Microsoft.Exchange.Services.OData.Model.Contact",
  "GivenName": "Santiago",
  "Surname": "Porras",
  "EmailAddress1": "s.porras@mvp.microsoft.com"
}
```

• Actualizar un Contacto:

Para modificar un contacto hay que enviar una solicitud "PATCH" hacia la dirección del contacto y se adjunta un objeto JSON con las propiedades que queremos modificar.

```
PATCH https://outlook.office365.com/ews/odata/Me/Contacts('sporras@mvp.microsoft.com')
Accept: application/json;odata.metadata=full
Content-Type: application/json;odata.metadata=full
Expect: 100-continue
```

```
{
  "Birthday": "1974-07-22",
  "Categories": [
    "MVP",
    "College Friend",
    "UX",
  ]
}
```

• Eliminar un Contacto:

Para eliminar un contacto, enviamos una petición DELETE para la dirección del contacto.

```
DELETE https://outlook.office365.com/ews/odata/Me/Contacts('sporras@mvp.microsoft.com')
Accept: application/json
```

Ejemplo: Utilizar estas APIs en una aplicación ASP NET. MVC

Para realizar este ejemplo, necesitaremos los siguientes requisitos previos:

- Tener instalado Visual Studio 2013 (versiones Profesional o Ultimate).
- Descargar las Office 365 API Tools bien directamente desde esta dirección <http://visualstudiogallery.msdn.microsoft.com/7e947621-ef93-4de7-93d3-d796c43ba34f> o bien desde Nuget.

Una vez tenemos instalada las Office 365 API Tools, abrimos Visual Studio y creamos un proyecto ASP. NET MVC 4 seleccionando la plantilla WebAPI. A continuación, a nuestro proyecto le vamos a añadir la conexión a los servicios REST de la API de Office 365. Para añadirlos, nos dirigimos al Explorador de soluciones, Agregar Servicio conectado tal y como se muestra en la siguiente imagen:

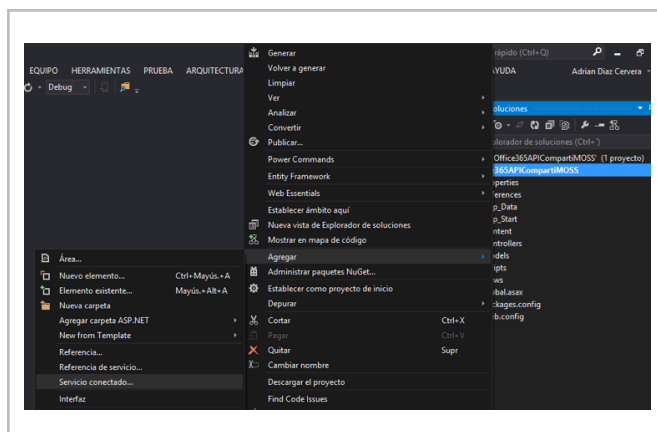


Imagen 1.- Añadir Servicio Conectado a nuestra App ASP.NET MVC.

Aparecerá el Administrador de servicios. Seleccionamos Office365 e introducimos los datos de usuario y contraseña de la cuenta de Office 365 que se vaya a utilizar. Una vez autenticado contra el Tenant de Office365, veremos una lista de tres servicios: Active Directory, Exchange y SharePoint. Inicialmente, la columna Permisos a la derecha de cada servicio estará vacía. En nuestro ejemplo vamos a seleccionar Exchange. Una vez seleccionado pulsamos Aceptar (Imagen 2).

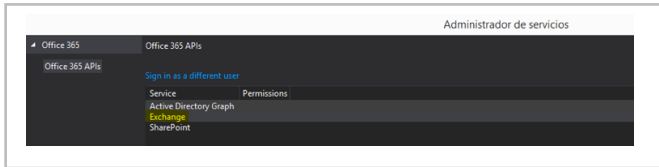


Imagen 2.- Administrador de Servicios.

Dado que no están establecidos los permisos, al tratarse de una versión preview, estos se solicitan por pantalla. En nuestro caso, vamos a otorgar todos los permisos para poder ver todas las opciones que vienen incluidas y que hemos visto en este artículo.

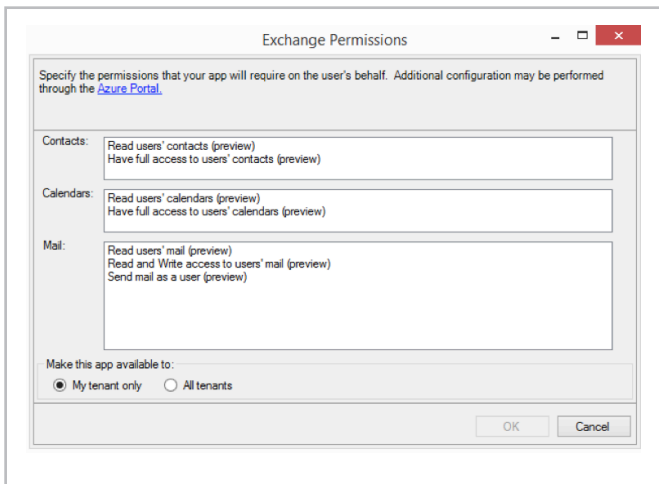


Imagen 3.- Permisos sobre las APIs de Exchange.

Una vez hemos añadido la API a nuestro proyecto revisamos la estructura de nuestro proyecto que quedará de la siguiente forma:

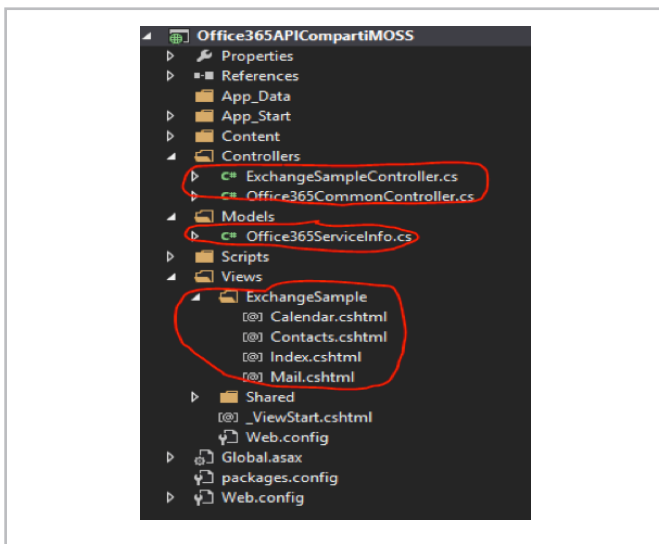


Imagen 4.- Estructura del proyecto en el explorador de soluciones.

Tal y como hemos remarcado en la imagen, se han añadido dos controladores (ExchangeSampleController, Office365CommonController), un modelo (Office365ServiceInfo) y una serie de vistas para poder comprobar las capacidades de estas APIs.

Si analizamos los ficheros que se han añadido:

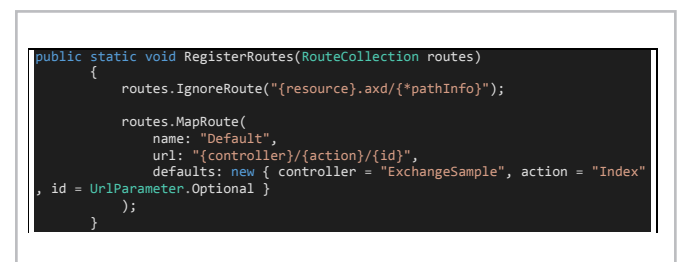
- Office365ServiceInfo: En esta clase disponemos del modelo que se utiliza para obtener la información para cada una de las APIs, para el caso de Exchange tendremos la siguiente información:
 - ResourceId = "https://outlook.office365.com/";
 - ApiEndpoint = "https://outlook.office365.com/ews/odata";
 - AccessToken = Office365CommonController.GetAccessToken(ResourceId);

- Office365CommonController: Se implementa un controlador común para manejar la autenticación y el control de errores sobre las peticiones al Tenant de Office 365. Además disponemos de diversos métodos, como por ejemplo, para obtener el token el usuario que esta logado en este momento e incluso para eliminar esta información del navegador (para evitar que nos puedan robar las credenciales y suplantarnos la identidad).

- ExchangeSampleController: Dentro del controlador de Exchange por un lado tendremos una serie de Clases: Calendario, Contactos, Mail para poder mapear el objeto JSON que nos devuelve la API en Clases .NET y poder mostrarlo en la vista. Por otro lado, dentro de este controlador tendremos tres métodos para obtener el Calendario, el Mail y los Contactos. Estos métodos no son más que peticiones HTTP a la API Rest correspondiente.

- Ficheros de Vistas: En estos ficheros están las vistas RAZOR para representar los datos obtenidos en las distintas peticiones a las APIs de Exchange.

Una vez analizado el código añadido, vamos a modificar el controlador por defecto, para ello dentro de nuestra solución nos dirigimos a la Carpeta "App_Start", editamos el fichero RouteConfig.cs. En esta clase tan solo tiene un método "RegisterRoutes", en el mismo modificamos el controlador por defecto, en nuestro caso "ExchangeSample", quedando de la siguiente forma:



Una vez realizada esta modificación, compilamos la solución y si todo ha ido correctamente visualizaremos la imagen 5

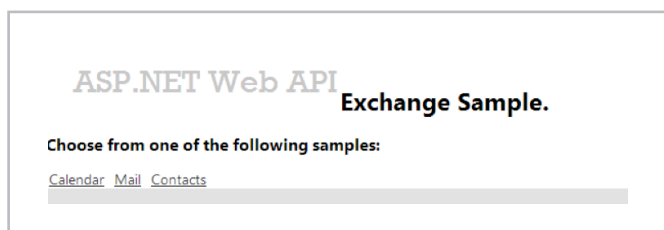


Imagen 5.- Ejemplo de uso de la API Exchange Online.

Al pulsar sobre cada opción: Calendar, Mail o Contact, se mostrarán las distintas vistas añadidas donde visualizamos la información solicitada a la API correspondiente.

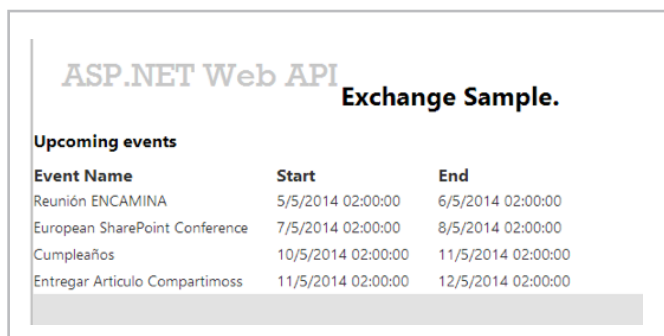


Imagen 6.- Resultados Obtenidos al pulsar sobre Calendar.

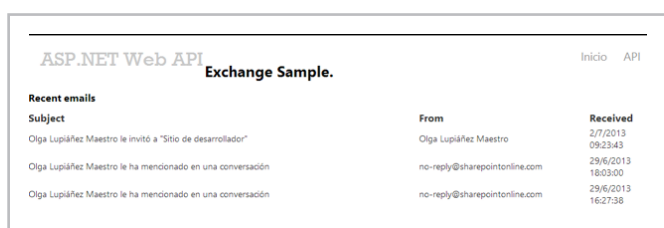


Imagen 7.- Resultados obtenidos al pulsar sobre Mail.

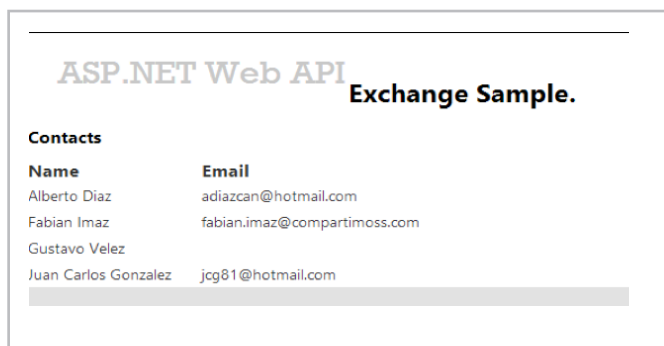


Imagen 8.- Resultados obtenidos al pulsar sobre Contact.

Dejo a la curiosidad de los lectores, añadir muchas más funcionalidades a este ejemplo base, como las operaciones CRUD sobre Calendar, Mail o Contacts.

Conclusiones

Office 365 es una plataforma que está creciendo cada vez más. Microsoft, piensa que el futuro es el "CLOUD" y que más pronto que tarde todas las empresas darán el salto. Personalmente, creo que hay productos que aún les falta cierta evolución para que las empresas se planteen dar el salto completo a la nube. Pero con novedades como las presentadas en Las Vegas, entre las que se encuentran estas APIs, cada vez hacen que la alternativa del cloud sea una opción más seria y más viable.

El poder utilizar las APIs desde aplicaciones fuera del entorno Office 365, es un gran adelanto y algo que facilitará la adopción de la plataforma. Hoy en día, hay muchas empresas que requieren de aplicaciones para el envío de mails, para consultar información del directorio activo, o incluso gestionar calendarios entre los integrantes de su organización. Ahora mismo, el obtener esta información mediante desarrollo es muy complejo y para solventar este inconveniente se planteaban escenarios híbridos. Pero esta solución es más cara y en algunos escenarios no dispones ni de las ventajas del cloud ni las ventajas de las infraestructuras propias. Con estas APIs todos estos requerimientos se podrán abordar de una forma simple y transparente.

En el siguiente número de CompartiMOSS desgranaremos las opciones que dispondremos para facilitar el trabajo con ficheros y carpetas en SharePoint/OneDrive como con el Active Directory.

Referencias:

- <http://msdn.microsoft.com/es-ES/library/office/dn605901>
- <http://msdn.microsoft.com/library/office/dn605893>
- <http://blogs.msdn.com/b/officeapps/archive/2014/03/12/announcing-office-365-api-tools-for-visual-studio-preview.aspx>

ADRIÁN DIAZ CERVERA

SharePoint Architect at Encamina

MVP SharePoint Server

<http://blogs.encamina.com/desarrollandosobresharepoint>

<http://geeks.ms/blogs/adiazcervera>

adiaz@encamina.com

[@AdrianDiaz81](https://twitter.com/AdrianDiaz81)

10

Cómo consumir los Servicios REST de SharePoint Online 2013 desde un WorkFlow en SharePoint Designer

Resumen

Desde SharePoint 2003 los desarrolladores hemos tenido acceso a funcionalidades de la plataforma a través de servicios web que se han expuesto para que las aplicaciones cliente, puedan ejecutar operaciones remotamente sobre la información disponible en los sitios de SharePoint.

Con la aparición de SharePoint 2007 y 2010, estos servicios web fueron mejorados y la cantidad de operaciones se amplió para permitir a los desarrolladores tener un mayor rango de acción sobre la plataforma a través de su uso. En el caso de SharePoint 2010 se introdujo una nueva interfaz y API que comenzaba a aprovechar la interfaz de transferencia de estado representacional (REST), por sus siglas en inglés.

Ahora con la llegada de SharePoint 2013 y su actualización al uso de estándares Web y nuevos protocolos, el espectro de opciones de desarrollo se amplió y es así como la plataforma mejoró la capa adicional que permite comunicar a las aplicaciones cliente a través de servicios, utilizando REST, la cual permite una mejor interacción de los desarrolladores con la información expuesta en los sitios de SharePoint. Esta capa ha sido ampliada y mejorada con respecto a SharePoint 2010, y actualmente los servicios web heredados de las versiones anteriores de la plataforma, ya se consideran obsoletos y no deberían ser utilizados.

En esta ocasión vamos a revisar los pasos necesarios para consumir los servicios REST que la plataforma expone, utilizando como aplicación cliente a SharePoint Designer 2013. Procederemos a construir un WorkFlow que ejemplifique y demuestre el llamado de una operación a través del uso de un servicio REST expuesto por la plataforma.

REST en la Plataforma SharePoint

El API REST que SharePoint 2013 expone es bastante grande, y es muy importante revisar la documentación para entender y tener una idea general de los diferentes servicios y su funcionalidad con los cuales contamos.

En el enlace [http://msdn.microsoft.com/en-us/library/office/jj860569\(v=office.15\).aspx](http://msdn.microsoft.com/en-us/library/office/jj860569(v=office.15).aspx) nos damos una idea de los distintos endpoints agrupados por los diferentes elementos de la plataforma, por ejemplo: listas, archivos y carpetas, usuarios y grupos, entre otros. En SharePoint 2010 para obtener vía REST las listas de un sitio de SharePoint se tenía que ejecutar el siguiente llamado a través del protocolo HTTP: http://intranet/_vti_bin/listdata.svc En ese caso listdata.svc es la interfaz REST a través del cual se realizan o ejecutan las distintas operaciones. En SharePoint 2013 se abrevió la llamada por intermedio del alias `_api`, y lo anterior se consigue de la siguiente forma: https://intranet/_api/lists

Fiddler el primer aliado

Fiddler es una herramienta bastante recomendada para ejecutar y probar las distintas operaciones REST que vayamos a utilizar desde un WorkFlow en SharePoint Designer 2013. Esto se debe a que la depuración y revisión rápida de lo que está sucediendo desde el WorkFlow y los pasos definidos en Designer, es realmente una tarea que consume demasiado tiempo y es difícil para el desarrollador que requiere realizar sus tareas con productividad. Por tal razón necesitamos descargar e instalar Fiddler (<http://www.telerik.com/download/fiddler>).

Es recomendable, si estamos trabajando desde Windows 8.1, descargar la versión específica de la herramienta para este sistema operativo, tal como lo aclaran en la página. A continuación veremos los pasos a seguir para ejecutar una operación REST de SharePoint 2013 desde Fiddler:

- Habiendo instalado Fiddler, ejecutamos la herramienta.
- Cuando cargue Fiddler utilizamos F12 para iniciar la captura del tráfico.
- En el lado derecho ubicar el TAB Composer, que es la ventana desde la cual se puede ejecutar las operaciones REST con la ventaja de poder indicar el método HTTP requerido como se muestra en la siguiente figura.

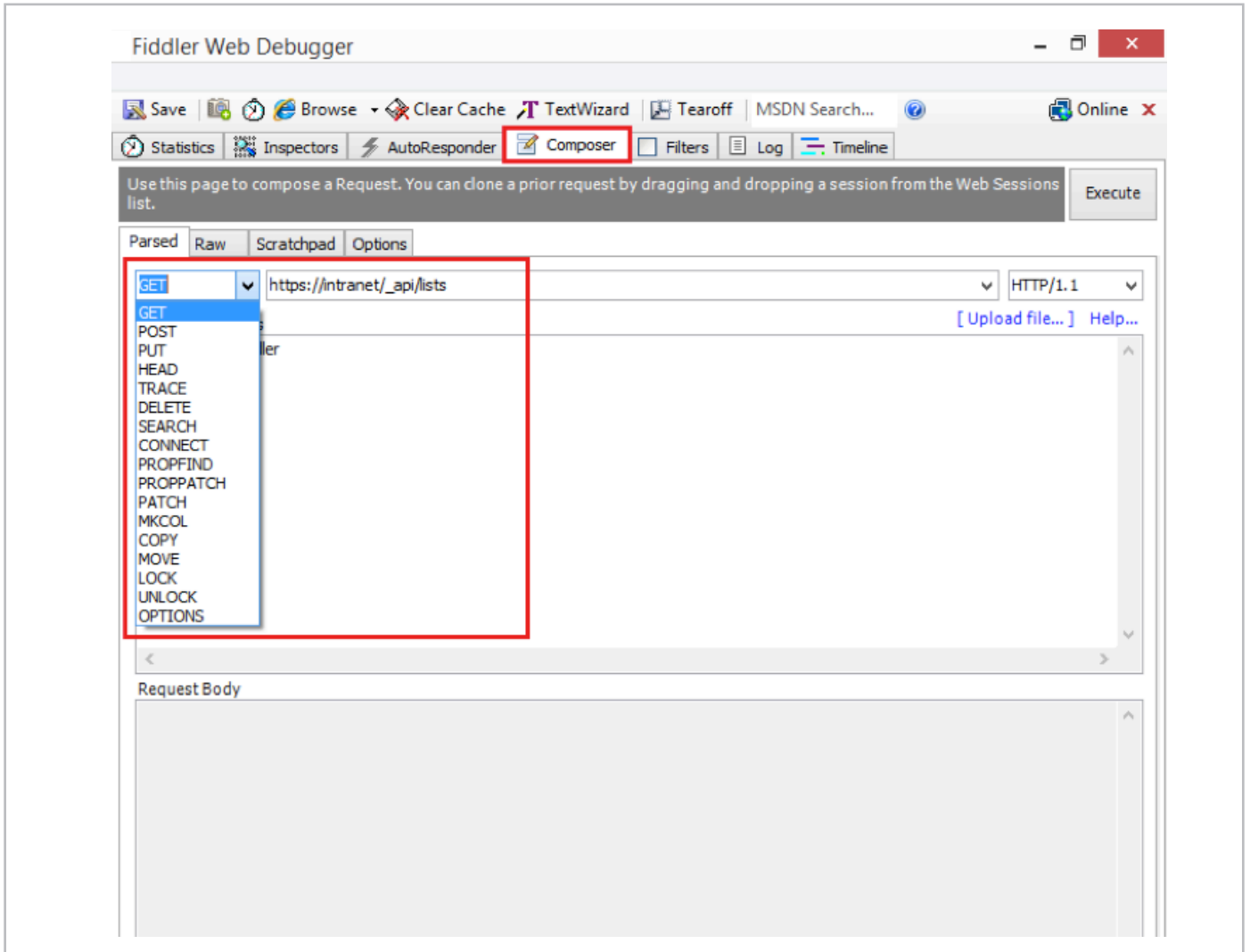


Imagen 1.- Muestra cómo componer una solicitud REST

- Al hacer clic en el botón Execute, vamos a observar diferentes problemas por la siguiente razón. En el caso de SharePoint Online 2013, que es la versión de SharePoint expuesta a través de los servicios de Nube de Microsoft Office 365, el método de autenticación utilizado es distinto de una implementación de SharePoint 2013 OnPremise. Por tal razón se requieren pasos adicionales en Fiddler para lograr ejecutar los llamados a la interfaz REST.
- Luego de ejecutar el servicio en Fiddler se puede ver un mensaje en color amarillo, sobre el cual se debe hacer clic para decodificar las respuestas que vienen desde el sitio de SharePoint. De inmediato se podrá ver un mensaje que nos indica que el llamado no se puede ejecutar por falta de permisos: HTTP/1.1 403 Forbidden.
- Es importante en el menú Tools de Fiddler hacer clic en Fiddler Options, seleccionar el TAB HTTPS y seleccionar las opciones que permiten capturar los llamados HTTPS, debido a que SharePoint Online 2013 hace las peticiones utilizando ese protocolo. Activamos las siguientes opciones como muestra la siguiente figura.

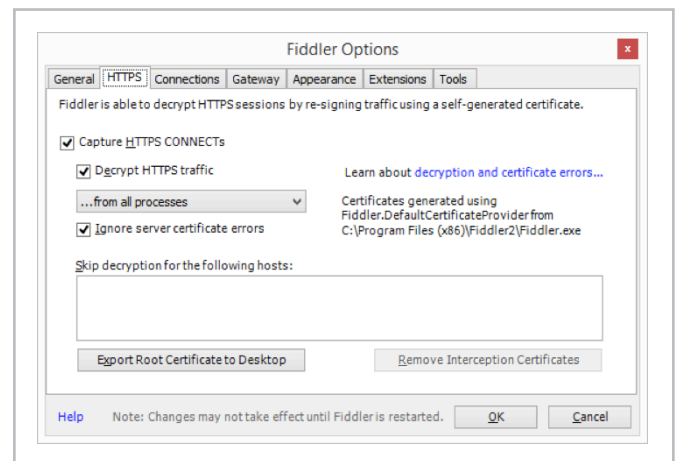


Imagen 2.- Muestra cómo activar la captura de las peticiones HTTPS.

- Como ya se tiene configurada la herramienta para escuchar las peticiones HTTPS, debemos abrir una ventana del navegador y ahí iniciar una sesión en el sitio de SharePoint Online 2013. En la ventana que solicita nuestras credenciales, ingresarlas, seleccionar la opción que permite mantener la sesión iniciada, y al finalizar la redirección al sitio, nuevamente regresamos a Fiddler para buscar las cookies que requerimos para poder hacer los llamados REST. Esto se puede buscar en Fiddler en la ventana que está escuchando todas las solicitudes realizadas y luego

encontrar el valor de la cabecera Cookie que se envían al sitio de SharePoint Online 2013 durante las peticiones. La siguiente figura muestra lo anterior.⁹



Imagen 3. Captura de los valores de Cookie utilizados para autenticar las peticiones en SharePoint Online 2013.

- Seleccionamos el valor FedAuth, y lo copiamos para luego utilizarlo.
- En Fiddler nuevamente regresamos al TAB Composer y agregamos la cabecera para realizar nuevamente la solicitud al servicio REST que habíamos intentado antes.
- En la siguiente figura se puede ver que en la ventana Request Headers hemos agregado el valor Cookie tal como se había copiado en un punto anterior. También es importante observar y agregar los valores User-Agent, Host y Content-Length. Al hacer clic en el botón Execute se podrá evidenciar que el servicio retorna una respuesta correcta: HTTP/1.1 200 OK

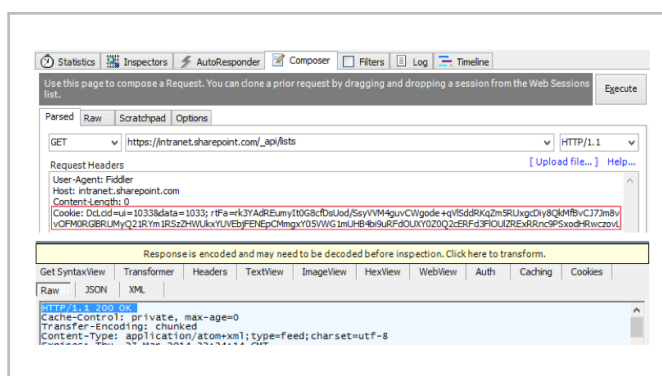


Imagen 4.- Muestra la configuración de la cabecera Cookie para componer la petición.

- En Fiddler en el TAB XML por ejemplo se puede visualizar los resultados retornados por la solicitud.

Configurando Fiddler para realizar solicitudes tipo POST

Hasta este punto hemos revisado la configuración y llamado a un servicio REST de SharePoint Online 2013 utilizando la operación GET, y también el envío de la cabecera Cookie

con los valores necesarios para que la ejecución sea permitida por el sitio de SharePoint. Ahora es momento de ejecutar una operación vía REST, utilizando la operación POST, la cual permite enviar información o una orden directamente al sitio de SharePoint y por ejemplo crear un sitio, dar de alta un documento, crear un ítem de una lista, entre otras operaciones. Para el propósito de este artículo, vamos a mostrar cómo podemos aprobar un documento en una biblioteca de documentos la cual ha sido configurada con la opción de Aprobación de contenido, haciendo uso de una operación vía REST, que ejecute la acción y deje el documento en el estado APROBADO. En el siguiente artículo se puede tener un detalle adicional de operaciones que se pueden ejecutar "Understanding and Using the SharePoint 2013 REST Interface" (<http://msdn.microsoft.com/en-us/magazine/dn198245.aspx>).

- Nuevamente regresamos a Fiddler y seleccionamos el TAB Composer.
- Esta vez vamos a configurar una operación tipo POST. La URL que invoca el servicio REST sería similar a lo siguiente: `https://intranet/_api/web/GetFileByServerRelativeUrl('/documentos/midocumento.docx')/approve(comment='El documento ha sido aprobado')`

La URL muestra que tenemos una biblioteca de documentos llamada DOCUMENTOS y un documento dado de alta con nombre MIDOCUMENTO.DOCX. El llamado a la operación `GetFileByServerRelativeUrl()` permite obtener una instancia del documento, el cual se pasa al método a través de su ubicación relativa en el sitio de SharePoint, no es necesario enviar la ruta absoluta del documento.

Luego se puede observar la ejecución de la operación

approve, la cual es la que permite realizar la APROBACIÓN del documento en cuestión, adicionando un mensaje cualquiera, tal como si se aprobara el documento desde la interfaz Web en el sitio de SharePoint.

NOTA: En caso que la biblioteca haya sido configurada para requerir CheckIn y CheckOut de los documentos, en el WorkFlow deberíamos ejecutar la acción que desproteja el documento para poder ejecutar acciones como la de APROBACIÓN y todo lo que requiera cambios sobre el elemento. Para este ejemplo, asumimos que no es necesario desproteger el documento.

- En Fiddler si ejecutamos el llamado REST ya descrito, vamos a obtener un Error 403 Forbidden, como se muestra en la figura.

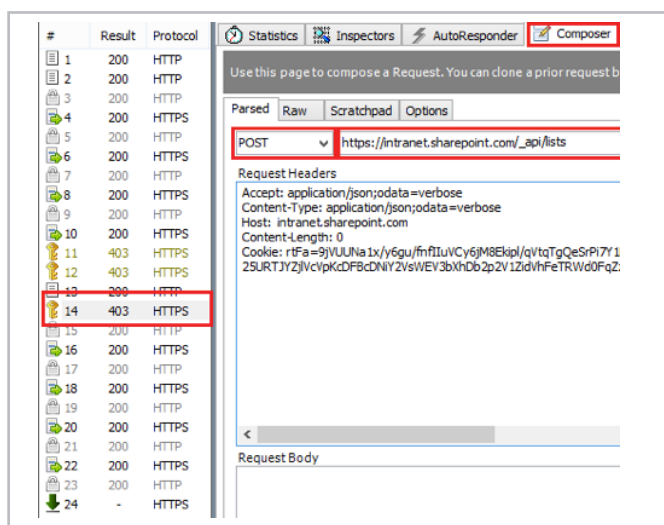


Imagen 5.- Muestra la configuración de una solicitud POST.

- El error es debido a la ausencia de un valor más en la cabecera de la solicitud. Ese valor es el del parámetro X-RequestDigest. Para conseguir dicho valor, es necesario componer una llamada a un servicio REST con la operación POST que retorna el valor de FormDigestValue, valor que se usará en la cabecera de solicitudes con la variable X-RequestDigest.
- La llamada se compone de la siguiente manera en el Composer de Fiddler: https://intranet.sharepoint.com/_api/contextinfo
Los valores de la cabecera de solicitudes deberán ser los mismos que hasta ahora se han usado.
- La llamada anterior a través de Fiddler permitirá capturar el valor de FormDigestValue. Ese valor se debe copiar y lo utilizamos como cabecera de la solicitud debiendo quedar algo similar a esto:

```
Accept: application/json;odata=verbose
Content-Type: application/json;odata=verbose
Host: intranet.sharepoint.com
Content-Length: 0
Cookie: rtFa=9jVUUNa1x/y6gu/fnflUVCy6jM8Ekipl/qVtqTgQeSrPi7Y1FBs4IzdfSYRPAyclniwAEPFpLT;FedAuth=77u/PD94bWwgdMvyc2ljbj0iMS4wIjBlbmNvZGluz0idXRmLTgiPz48U1A+RmFsc2UZXMvYWY1cmJjYXNkbXMvc3Rvcms8L1NQPg==;
X-RequestDigest: 0x191DC189CFA557711C8630CD0E3E6DBD601267159D2177AD4CFF3AE45D248F9CF9B7E7,27 Mar 2014 21:55:51 -0000
```

- Teniendo el valor de X-RequestDigest es momento de volver a ejecutar el servicio REST que nos permitirá APROBAR el documento: [https://intranet/_api/web/GetFileByServerRelativeUrl\('/documentos/midocumento.docx'\)/approve\(comment='El documento ha sido aprobado'\)](https://intranet/_api/web/GetFileByServerRelativeUrl('/documentos/midocumento.docx')/approve(comment='El documento ha sido aprobado'))

La llamada anterior desde Fiddler deberá retornar un resultado 200 OK. Y si se revisa el documento en el sitio de SharePoint este deberá aparecer como APROBADO con el mensaje "El documento ha sido aprobado".

Crear y configurar un WorkFlow en SharePoint Designer 2013

Luego de probar y validar el correcto funcionamiento de nuestras solicitudes REST desde Fiddler, es el momento de crear el WorkFlow en SharePoint Designer, agregar la acción que permite invocar el servicio REST y tenemos una alta probabilidad que todo funcione de manera correcta.

La acción Registrar en lista de historial es una buena forma de registrar valores que nos permiten ir validando que el WorkFlow se está ejecutando de manera adecuada así como los servicios REST invocados.

- Lo primero es que debemos contar con permisos necesarios para realizar los siguientes pasos, por ejemplo ser Propietarios del sitio.
- Teniendo los permisos suficientes, procedemos a abrir SharePoint Designer 2013 y cargamos el sitio en el cual vamos a crear el WorkFlow.
- De inmediato vemos que se cuenta con varias opciones para crear WorkFlows: WorkFlow de lista, reutilizables y de sitio. Para el propósito de este ejemplo, usaremos el WorkFlow de Lista y seleccionamos la lista o biblioteca de documentos sobre la que queremos que se ejecute el WorkFlow.
- Luego se abrirá un Wizard que nos permitirá configurar las opciones iniciales del WorkFlow. Por ejemplo asignarle un nombre, una descripción, y lo más importante escoger el tipo de Plataforma, en este caso y para tener la opción de invocar servicios REST desde el WorkFlow, debemos seleccionar SharePoint 2013 WorkFlow.

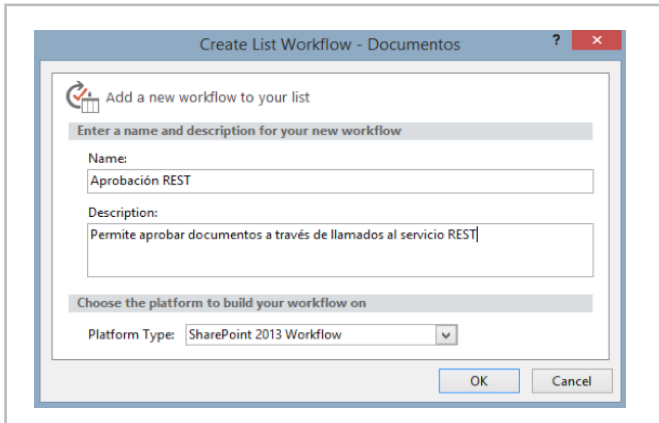


Imagen 6.- Creación de un WorkFlow de tipo Lista en SP Designer 2013.

- Al hacer clic en el botón OK, SharePoint Designer 2013 despliega una ventana que permitirá crear y configurar los distintos pasos del WorkFlow.
- A continuación asignamos un nombre a la fase, por ejemplo Aprobación. En la sección Transition to stage hacemos clic y agregamos la acción Ir a una fase y seleccionamos el valor Final del flujo de trabajo. En este caso es un ejemplo simple que no requiere de varios pasos, pero podemos agregar todos los pasos o fases necesarias para nuestro proceso.
- En la fase de Aprobación vamos a agregar una acción que invoque al servicio REST que nos permite obtener la información de contexto de la cual como ya vimos desde Fiddler, obtenemos el valor de FormDigestValue. La acción de SharePoint Designer 2013 que nos permite hacer esto es: Llamar al servicio web HTTP. Las acciones de los WorkFlows de SharePoint Designer pueden contener valores estáticos, como lo veremos en este ejemplo, pero también podemos trabajar con valores dinámicos, por ejemplo tomados desde la metadata del documento, de parámetros o variables definidos en el WorkFlow, y otras opciones más que se tienen a disposición.
- Lo primero que la acción nos solita es que configuremos la URL del servicio y también el método HTTP, tal como lo hicimos en Fiddler.

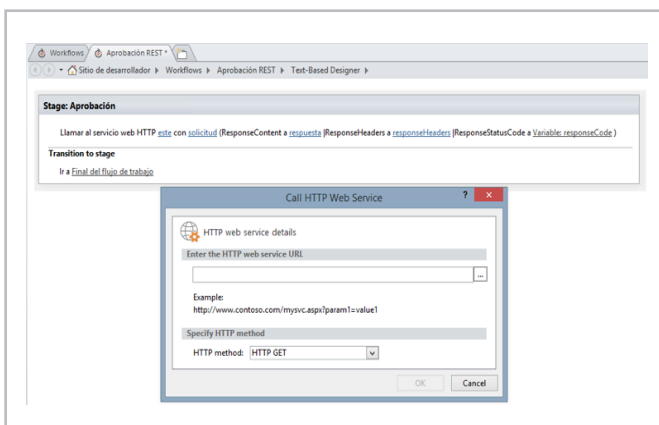


Imagen 7.- Configuración de la fase y acciones del WorkFlow.

- En el campo que nos permite componer la URL agregamos la que habíamos utilizado en Fiddler y el método requerido. Para este caso la URL es similar a lo siguiente: https://

intranet/_api/contextinfo y usamos el método HTTP POST.

- Ahora la pregunta es en qué momento pasamos los valores a la cabecera de la solicitud, tal como lo hicimos en Fiddler. Lo primero que tenemos que hacer es agregar una acción antes del llamado al servicio que nos permite configurar un Diccionario. Agregamos la acción Generar Diccionario y comenzamos a configurar cada uno de los valores de la cabecera, y deberíamos tener algo similar a lo siguiente.

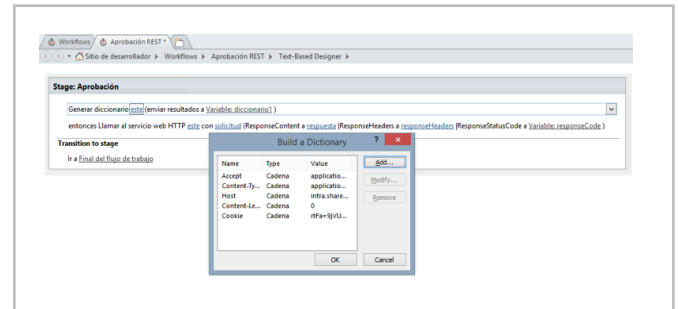


Imagen 8.- Configuración de la acción Generar Diccionario.

- Finalizamos la configuración del diccionario haciendo clic en el botón OK. Por defecto SharePoint Designer crea la variable diccionario, pero en la acción se puede crear una nueva si así se desea.
- Luego hacemos clic derecho sobre la acción Llamar al servicio web HTTP que ya habíamos agregado y seleccionamos la opción Propiedades. En la ventana que se abre ubicamos la opción RequestHeaders y ahí asignamos como valor el diccionario que acabamos de configurar en pasos anteriores.
- Con lo anterior el servicio podrá ejecutarse correctamente y es momento de obtener el valor de FormDigestValue retornado por el servicio. Para esto creamos una variable local con el nombre FormDigestValue o el que se desee, de tipo Diccionario.
- En la acción Llamar al servicio web HTTP debemos hacer clic en el enlace respuesta. Ahí es donde asignamos la variable local recién creada en la cual se almacenará el valor de FormDigestValue.
- El servicio almacenará el valor de retorno en la variable de tipo diccionario, y su valor puede ser obtenido agregando una acción Obtener un elemento de un diccionario. El servicio retorna la información en formato JSON y en el enlace de la acción Elemento por nombre o ruta de acceso se puede obtener el valor JSON de la siguiente manera: d/GetContextWebInformation/FormDigestValue

Se debe indicar también la variable diccionario de la cual se toma el valor y el resultado leído deberá ser almacenado en una variable adicional que puede ser creada al hacer clic en el enlace elemento. Esta variable adicional será de tipo cadena. En la siguiente imagen se observa lo que debemos tener hasta este momento.

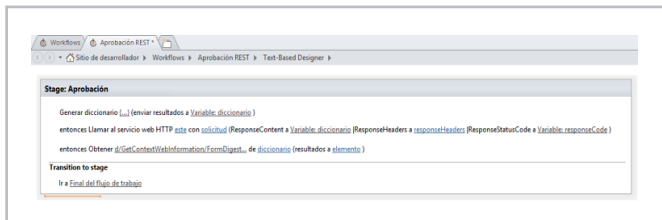


Imagen 9.- Se obtiene el valor retornado por el servicio para FormDigestValue.

- Con el valor de FormDigestValue es posible agregar la acción que permitirá hacer la llamada al servicio REST que nos permitirá APROBAR el documento.

- En pasos anteriores habíamos definido una variable de tipo Diccionario con los parámetros y sus valores que debemos pasar como cabecera de las solicitudes. Pero es necesario agregar el valor de la variable donde se almacenó el valor de FormDigestValue a este diccionario. Esto se puede hacer agregando una acción de tipo Generar Diccionario a través de la cual asignamos el valor a un nuevo parámetro llamado X-RequestDigest, que como ya se explicó en Fiddler, es requerido para poder ejecutar la operación POST que nos permitirá APROBAR el documento. La acción debe ser configurada usando la variable de tipo Diccionario que ya se había utilizado cuando se agregaron los parámetros requeridos por los servicios.

- Seguidamente agregamos una nueva acción de tipo Llamar al servicio web HTTP, asignamos la URL del servicio tal como se usó en Fiddler, y para efectos del ejemplo usaremos valores prestablecidos, pero es importante recordar que estos valores pueden ser dinámicos. Esto es importante porque el servicio debería APROBAR cualquier documento que esté siendo asociado al WorkFlow, igualmente la ubicación de la biblioteca y cualquier elemento que sea susceptible de ser dinámico y pueda ser construido dentro de la URL del llamado al servicio. Por ahora nuestra URL luce así:

https://intranet/_api/web/GetFileByServerRelativeUrl(\"/documentos/midocumento.docx\")/approve(comment='El documento ha sido aprobado'). Recordemos que el método HTTP debe ser configurado como POST.

Nuevamente hacemos clic derecho sobre la acción y asociamos en el valor RequestHeaders la variable Diccionario que hemos venido utilizando para los parámetros requeridos por la cabecera de las solicitudes y que en el paso anterior fue actualizada con el valor de X-RequestDigest.

- Luego de haber seguido los pasos anteriores debemos tener el WorkFlow similar a como se ve en la siguiente figura.

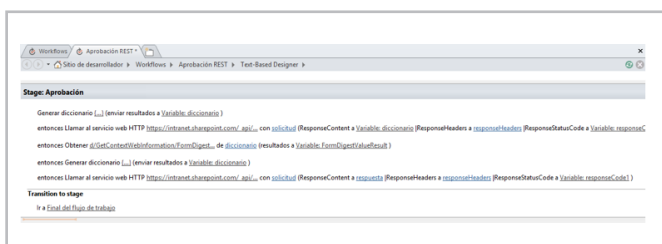


Imagen 10.- Configuración final del WorkFlow

- Para finalizar la configuración del WorkFlow hacemos clic en su nombre, Aprobación REST, desmarcamos el valor Automatically update the workflow status to the current stage name y seleccionamos la opción Start workflow automatically when an item is created.

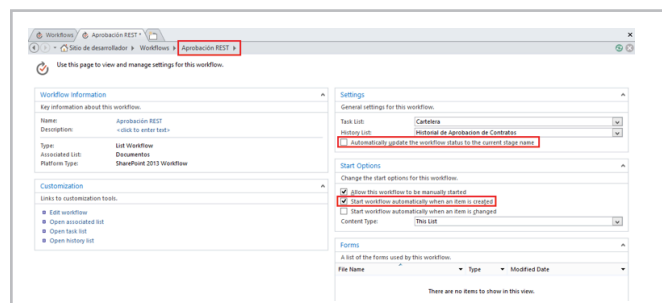


Imagen 11.- Configuración final del WorkFlow.

- Finalmente hacemos clic en el botón Guardar y Publicar de SharePoint Designer 2013 para poder probar el WorkFlow en la biblioteca de documentos.

- Para probar el WorkFlow vamos al sitio de SharePoint, ubicamos la biblioteca a la cual asociamos el WorkFlow que se acaba de crear y configurar, damos de alta un documento a la biblioteca y el WorkFlow deberá iniciar y terminar satisfactoriamente. Luego de esto al revisar el historial de versiones del documento o las opciones de Aprobación, debemos encontrar que ha sido aprobado con el mensaje que hemos usado en este ejemplo.

En conclusión hemos podido evidenciar que la inclusión y uso de REST dentro de SharePoint, es una gran ventaja para que los desarrolladores puedan consumir y enviar información a los sitios en SharePoint desde las aplicaciones cliente que implementan, y que se encuentra disponible en las versiones Online y Onpremise de la plataforma. Debido al uso de REST los desarrolladores de gran variedad de plataformas podrán construir estas aplicaciones que se comunican con SharePoint: JAVA, PHP, Android, IOs, entre otras.

Finalmente vemos que la inclusión de la acción que permite hacer llamado a los servicios REST de SharePoint desde los pasos de un WorkFlow, es una capacidad realmente potente e importante para que el resultado final de los WorkFlows construidos con Designer realmente agreguen valor a las organizaciones que muchas veces requieren integrar estos procesos con información adicional, y no estar atados únicamente a los ítems de una lista y/o biblioteca.

ANDRÉS ORTÍZ

Microsoft Community Specialist SharePoint

aortiz@topgroup.com.ar

@jaortizgonz

http://desarrollo.topgroup.com.ar/search/label/Sharepoint

16

Ajuste de la Relevancia en el Motor de Búsqueda de SharePoint 2013

Resumen

Después de instalar y configurar el Motor de Búsqueda de SharePoint 2013 correctamente, es indispensable ajustar la relevancia de los resultados de acuerdo a las condiciones específicas requeridas por el sistema que lo utiliza.

Introducción

Cuando se instala y configura correctamente el Motor de Búsqueda de SharePoint, los resultados que genera para una consulta aparecen en el orden que los algoritmos internos determinan. Pero el orden por defecto generado por SharePoint no es necesariamente el orden que los requisitos de la empresa o compañía que utiliza el sistema desean tener. Un ejemplo común es que puede exigir que los productos de la empresa aparezcan en primer lugar en los resultados de búsqueda.

La relevancia se puede definir como los resultados que elevan la probabilidad de realizar una tarea con éxito. Aplicado a un sistema de búsqueda de información, se puede decir que los resultados más relevantes producidos por una búsqueda son los que con más probabilidad presentan la información que el usuario está buscando. En el ejemplo indicado en el párrafo anterior, es mucho más probable que un empleado de una empresa busque información sobre los productos de su propia empresa, por lo que los resultados más relevantes son los que están relaciones con estos productos, y no con los de la competencia.

El Motor de Búsqueda de SharePoint 2013 tiene algoritmos para calcular la relevancia ("Ranking") que son basados en modelos lingüísticos generales. Idealmente, los resultados mostrados en la primera página son los que contienen la información que el usuario está buscando. Cuando esto no ocurre, se habla de que es necesario ajustar la relevancia. Para esto, SharePoint proporciona los mecanismos necesarios para alterar los modelos y adaptar la relevancia a las necesidades particulares de un sistema.

Relevancia en el Motor de Búsqueda de SharePoint 2013

SharePoint utiliza dos tipos de fórmulas de ranking para calcular la relevancia: Estáticas y Dinámicas. La diferencia es relativa a si el cálculo del ranking se ve afectado o no por

los términos de la consulta y las propiedades del elemento de contenido. El ranking Dinámico describe el ranking que se ve afectado por los valores de las propiedades del contenido (también conocido como "query-dependent ranking"). Los componentes principales usados por el algoritmo Dinámico de ranking en el Motor de Búsqueda de SharePoint para calcular la relevancia son:

- **Texto de Anclaje ("Anchor Text")** - Es el texto que se incluye con un hyperlink para describir el contenido del enlace. Cuando el Motor de Búsqueda rastrea el elemento, este texto es incluido en el índice del ítem. El Anchor Text solamente influye en el rango, y no es un factor determinante para incluir un elemento en el grupo de resultados. Por ejemplo, si los términos de búsqueda se encuentran solamente en el Anchor Text y no en el contenido real del elemento, el vínculo puede ser obsoleto de tal forma que el contenido del elemento no se incluye en el grupo de resultados.
- **Pesaje de Propiedades ("Property Weighting")** - Algunas propiedades son más importantes para calcular la relevancia que otras. El Motor de Búsqueda de SharePoint provee una forma para modificar el "peso" de cada propiedad de tal forma que tengan más importancia en los cálculos de relevancia. Cambiar arbitrariamente esta propiedad puede tener efectos adversos en la relevancia general del sistema, por lo que no es recomendado hacerlo sin una evaluación de los cambios y como impactan la precisión de los resultados.
- **Normalización de la Longitud ("Property Length Normalization")** - Un elemento de contenido puede tener propiedades de diferente longitud. Si los valores de esas propiedades se tratan igualmente sin importar su tamaño durante los cálculos de relevancia, puede tener un efecto negativo sobre el ranking calculado. "Length Normalization" ajusta el rango de un elemento, basado en la longitud de la propiedad y su configuración de "Length Normalization".
- **Igualación de URL ("URL Matching")** - URL Matching es el proceso por el cual el Motor de Búsqueda controla la URL de un elemento para que corresponda con el término buscado por el usuario.

- **Extracción de Títulos (“Title Extraction”)** - Title Extraction, o utilizar el valor del título en el cálculo de la relevancia, puede ayudar a devolver contenido altamente relevante si el elemento está titulado de forma apropiada.

A su vez, el ranking Estático describe el ranking que no es impactado por el contenido o valor de las propiedades de un elemento (también conocido como “query-independent ranking”). Los componentes usados por este algoritmo son:

- **Distancia de cliqueo (“Click Distance”)** - Un documento, una página web, una lista o cualquier otro elemento puede estar vinculado a otro elemento porque el elemento vinculado contiene información relacionada a, y enriquece el contenido del elemento original que contiene el vínculo. Además, información sobre esos hyperlinks a un elemento específico, tales como el número de hyperlinks a él o en donde esos hyperlinks están localizados son importantes para determinar la relevancia. Click Distance se refiere al número de vínculos entre el elemento y una página “principal”. Para calcular la relevancia, el punto de inicio es esta página principal (“Authoritative Page”). Entre más vínculos el rastreador tenga que recorrer desde la página principal al elemento, menor es la relevancia. Si hay múltiples rutas al elemento, la relevancia es calculada en base al camino más corto, el que tiene la menor cantidad de links desde la página autoritativa hasta el elemento.
- **Profundidad de URL (“URL Depth”)** – El contenido importante o relevante está frecuentemente localizado más cerca al inicio de la jerarquía de sitios que en un sitio varios niveles más abajo. La consecuencia es que un URL más corto es probablemente mejor recordado y utilizado por un usuario. El Motor de Búsqueda revisa la profundidad del URL calculando el número de slash (“/”) en el URL. Entre mayor cantidad de slash, más profundo es el URL y menor la relevancia del elemento.
- **Detección automática de lenguaje (“Automatic Language Detection”)** – Los usuarios normalmente prefieren recibir contenido en su propio lenguaje. El Motor de Búsqueda determina el lenguaje del usuario basado en la cabecera “Accept-Language” del navegador.

Ajuste de la Relevancia en SharePoint 2013

Aunque hay muchos factores que determinan la calidad de los resultados de una búsqueda, comenzando por supuesto por la consulta misma utilizada por el usuario, no todos los factores que contribuyen al cálculo de la relevancia pueden ser modificados. De todas formas, en SharePoint 2013 es posible manipular la relevancia a través del uso de diccionarios, páginas relevantes, reglas

de ordenamiento dinámico y modelos de ranking.

Diccionarios de Búsqueda

Aunque estrictamente hablando los diccionarios no permiten modificar la relevancia en el Motor de Búsqueda, si es posible mejorar la experiencia de uso de las búsquedas influyendo indirectamente en la relevancia. Por ejemplo, definiendo diccionarios de sinónimos se puede lograr que los resultados se ajusten mejor a lo esperado.

Los tipos de Diccionarios de Búsqueda y su creación y uso es un tema demasiado extenso para este artículo, por lo que la segunda parte a publicar en el número 21 de CompartiMOSS se dedicará exclusivamente a los Diccionarios.

Páginas Relevantes

Las Páginas Relevantes son páginas que vinculan a información crítica dentro de la granja de SharePoint. Cuando se definen Páginas Relevantes, se afecta el ranking de cada página que está conectada por vínculos a la página Relevante por medio de “click distance”. “Click distance”, como se definió anteriormente es el número de clics necesario para navegar desde la Página Relevante al contenido a rastrear. Entre más cerca esté la información a la Página Relevante, su ranking estático es mayor.

Las Páginas Relevantes se definen en la Aplicación de Servicio de Búsqueda (“Search Service Application”, SSA) en donde se pueden definir tres niveles de Páginas Relevantes lo mismo que Páginas que no son Relevantes (las que ayudan a eliminar información irrelevante del cálculo de relevancia). En SharePoint On-Premise se define como Páginas Relevantes por defecto la página principal de la colección de sitio de más alto nivel en la Aplicación Web. Es obligatorio tener por lo menos una página Relevante definida. Normalmente los administradores de SharePoint pueden definir algunas otras Páginas Relevantes, como por ejemplo la página de home de un subsitio, aunque se pueden definir múltiples si es necesario.

Los Administradores con conocimientos específicos de un área pueden influenciar la importancia relativa de páginas especificando otras páginas Relevantes y no-Relevantes. Un ejemplo de Páginas no-Relevantes podrían ser los URLs de sitios que contienen información obsoleta que solamente se mantiene como archivo.

Para acceder al menú de configuración de páginas Relevantes, desde la Administración central de SharePoint vaya a las páginas de Administración del Búsquedas y en el menú de acceso rápido (lado izquierdo) se encuentra el vínculo hacia “Páginas relevantes”.

Páginas web relevantes

Las páginas relevantes son aquellas que se vinculan con la información más relevante. La función de búsqueda usa esta lista para calcular el rango de cada página del índice. Es posible que haya muchas páginas web relevantes en su entorno. Escriba las más valiosas en el cuadro de texto superior. También tiene la opción de especificar páginas relevantes de segundo y tercer nivel en los cuadros de texto inferiores.

Escriba una dirección URL por línea. Ejemplo: `http://intranet/directorio_de_sitios.html`.

Páginas más relevantes: *

`http://spses/`

Imagen 1.- Configuración de Páginas Relevantes.

SharePoint contiene todos los cmdlets de PowerShell necesarios para trabajar con páginas Relevantes:

- **Get-SPEnterpriseSearchQueryAuthority** - Obtiene una página Relevante o la colección de páginas Relevantes en el SSA.
- **New-SPEnterpriseSearchQueryAuthority** - Crea una nueva página Relevante.
- **Set-SPEnterpriseSearchQueryAuthority** - Cambia el nivel de una página Relevante.
- **Remove-SPEnterpriseSearchQueryAuthority** - Elimina una página Relevante existente.
- **Get-SPEnterpriseSearchQueryDemoted** - Obtiene las páginas no Relevantes ("no autoritativas").
- **New-SPEnterpriseSearchQueryDemoted** - Crea una página no Relevante.
- **Remove-SPEnterpriseSearchQueryDemoted** - Elimina una página no Relevante.

Los parámetros que estos cmdlets utilizan son:

- **Url** - Los comandos "New" necesitan un Url para designar la página o site que se va a agregar.
- **Identity** - [optional para los comandos "Get"] Sin una Identidad, el comando Get obtiene todas las páginas Relevantes. Los comandos "Set" y "Remove" requieren que una página específica sea el identificador.
- **Level** - Especifica el nivel al que la página se va a agregar. Valores válidos son 0, 1 y 2, donde 0 es la página más Relevante y 2 la de tercer nivel.
- **Owner** - Tiene que ser asignado a un "Search Object Owner" designando o la SSA (Search Service Application) o la Suscripción del Inquilino ("Tenant Subscription") en un sistema Multi-tenant.
- **SearchApplication** - El objeto SSA al que se le va a configurar las páginas.

El siguiente ejemplo crea una página Relevante en el segundo nivel, luego la mueve al tercero, crea una página no-Relevante y finalmente elimina la página que se movió al tercer nivel:

```
# La cuenta utilizada por el Motor de Búsqueda
$myPropietario = Get-SPEnterpriseSearchOwner
-Level ssa
```

```
# Crear una nueva página en el segundo nivel
$mySqa = New-SPEnterpriseSearchQueryAuthority
-Owner $myPropietario -SearchApplication $ssa -Url
"http://servidor/sitios/abc" -Level 1

# Mover la página del segundo al tercer nivel
Set-SPEnterpriseSearchQueryAuthority -Identity
$mySqa -Level 2 -SearchApplication $ssa -Owner
$myPropietario

# Crear una nueva página como no-autoritativa
New-SPEnterpriseSearchQueryDemoted -Owner
$myPropietario -SearchApplication $ssa -Url "http://
servidor/sitios/noautoritativa"

# Remover la página que se movió al tercer nivel
Remove-SPEnterpriseSearchQueryAuthority
-Identity "http://servidor/sitios/abc" -Owner
$myPropietario -SearchApplication $ssa
-Confirm:$false
```

Reglas de Consulta y Ordenamiento Dinámico

Para manipular la relevancia de los resultados de una consulta es posible establecer reglas de dos tipos: reglas que ejecutan antes de que se genere el grupo de resultados (Reglas de Consulta) y reglas que ejecuten una vez se tienen los resultados (Reglas de Ordenamiento). En ambos casos se puede lograr que determinados resultados aparezcan más hacia el comienzo de la lista (promover resultados) o que se muestren como menos importantes, moviéndolos hacia puestos menores en la lista (degradar resultados).

Aunque el resultado de ambos tipos de reglas puede llegar a ser similar, el primero se integra profundamente con los algoritmos internos de ranking del Motor de Búsqueda, lo que mejora el rendimiento de búsqueda y muestra de resultados. El segundo tipo de reglas se basa en un reordenamiento de los resultados ya rankeados por el Motor, proceso que teóricamente puede mostrar algún tipo de degradación en el rendimiento, aunque solo sea marginal.

Reglas de Consulta Dinámicas

Sin utilizar ningún tipo de programación, los

administradores de SharePoint, los administradores de Colecciones de Sitios y los propietarios de Sitios pueden modificar la relevancia por medio de “Reglas de Consulta” (“Query Rules”). En una Regla de Consulta se definen las condiciones de la Regla y las acciones que ocurren cuando una condición ocurre reduciendo la cantidad de resultados o cambiando el orden en que se muestran. Por ejemplo, una condición de Regla de Consulta puede ser que un término en la consulta corresponda con una (o unas) palabra determinada en un Conjunto de Términos de SharePoint (definido en el Servicio de Metadatos Administrados) o que es una consulta realizada frecuentemente (tal como “video”, por ejemplo). Cuando la condición ocurre, se puede iniciar una acción para mostrar un elemento específico al principio de la lista de resultados.

Como se indicó anteriormente, las Reglas de Consulta se pueden crear a nivel de Administración Central, Colección de Sitios o Sitios por medio de las páginas de administración de cada nivel. Las Reglas se heredan de un alcance superior al siguiente, pero cada nivel puede tener sus propias Reglas que no afectan a su nivel superior (pero son heredadas hacia su nivel inferior). SharePoint dispone de un número de Reglas predefinidas por defecto, pero su número depende de la licencia utilizada, haciendo por ejemplo que algunos tipos de Reglas no se puedan utilizar en SharePoint Foundation, pero si en SharePoint Server.

Existen seis tipos de condiciones para Reglas de Consulta:

- La consulta coincide exactamente con la palabra clave - Es la forma más sencilla de condición: la regla asociada se dispara cuando la palabra o frase configurada se utiliza exactamente en la consulta de búsqueda.
- La consulta contiene el término de acción - Similar a la anterior, con la excepción que la palabra o frase configurada esta al principio o al final de la consulta. Por ejemplo, si se configura “SharePoint” como condición, la acción iniciara con las consultas “SharePoint como sistema” y “información sobre SharePoint”, pero no con la consulta “información sobre SharePoint como sistema” o simplemente “SharePoint”. En esta condición es posible utilizar palabras o frases configuradas en el Servicio de Metadatos Manejados.
- La consulta coincide exactamente con el diccionario - Es igual al primer tipo de consulta, excepto que no se utiliza una palabra o frase libremente configurable, sino términos guardados en el Servicio de Metadatos Manejados. Hay que tener en cuenta que el Motor de Búsqueda chequea cada 30 minutos por cambios en los diccionarios guardados en el Servicio de Metadatos Manejados, por lo que si estos se modifican, los cambios tardarán algunos minutos en ser visibles.
- Consulta más común en este origen - La Regla de Consulta

se activa si la consulta se ha realizado con frecuencia en otro Origen de Resultados (permite seleccionar el Origen a utilizar).

- Tipo de resultados seleccionados con más frecuencia - Dispara la acción si en los usuarios han utilizado con frecuencia resultados que aparecen en consultas similares.

- Coincidencia de texto de consulta avanzada - Para utilizar Expresiones Regulares.

A su vez, tres tipos de acciones se pueden acoplar a las condiciones:

- Resultados promocionados - Es un resultado que aparece en la parte superior de la lista de resultados y que es muy importante que sea visible para el usuario. En versiones anteriores de SharePoint se llamaba “Best Bet” o “Visual Best Bet”. El ejemplo típico de utilización es definir la condición como “La consulta coincide exactamente con la palabra clave” configurando el nombre de la empresa, y como Resultado promocionado la información de contacto de ella. De esta forma, cuando un usuario utiliza el nombre de la empresa como termino de consulta, el primer resultado en la lista de resultados indicará directamente su información de contacto

- Bloques de resultados - Todos los resultados que cumplen la condición son agrupados en un bloque y resaltados dentro del grupo de resultados de la consulta. El bloque se puede mostrar al principio de la lista de resultados, o dentro de ella dependiendo de su ranking. Configuraciones adicionales permiten especificar cuantos resultados se deben mostrar en el bloque y si se debe mostrar un vínculo con “Más resultados” si la cantidad de resultados en el bloque sobrepasa la cantidad a mostrar configurada. Importante en este tipo de acciones es que permite mezclar resultados provenientes de diferentes Fuentes de Resultados, como por ejemplo, OpenSearch, Exchange y sistemas de SharePoint remotos, con los resultados nativos del sistema

- Cambiar resultados clasificados cambiando la consulta - Esta posibilidad permite cambiar la forma en la que una consulta es manejada por SharePoint, suplementándola con términos o Propiedades Manejadas adicionales

Finalmente es posible definir una fecha de inicio y finalización de publicación de la Regla, lo mismo que dejarla definida pero inactiva y una cuenta de usuario que recibe un mensaje de advertencia para revisar la regla en una fecha determinada.

Como un ejemplo de configuración utilizando la Interfaz de Usuario, vaya a la “Configuración del sitio” desde el sitio de nivel superior de una Colección de Sitios - “Reglas de consultas de búsqueda” (sección “Administración de la colección de sitios”) y en la página de “Administración de reglas de consulta” seleccione “Documentos (sistema)”

en el primer combo-box. Luego utilice el vínculo de “Nueva regla de consulta”, indique un nombre para la regla y seleccione “Consulta coincide exactamente con la palabra clave”, indicando “sharepoint documento” en la expresión. Utilice el vínculo de “Agregar resultado promocionado” e indique un título a mostrar en la lista de resultados, lo mismo que el URL a donde debe redirigirse cuando se hace clic sobre él. Se puede utilizar el campo de Descripción para agregar información adicional en la lista de resultados:

Imagen 2.- Definición de un resultado promocionado.

Guarde la Regla y ejecute una búsqueda utilizando el término exacto configurado. Al principio de la lista de resultados aparecerá el término promocionado:

Imagen 3.- Ejemplo de resultados promocionados.

SharePoint 2013 no dispone de cmdlets de PowerShell para trabajar con Reglas de Consulta. Aunque el Modelo de Objetos de Servidor contiene el espacio de nombres Microsoft.Office.Server.Search (Microsoft.Office.Server.Search.dll en el hive 15/ISAPI) que si debería permitir trabajar programáticamente con Reglas de Consulta, una gran parte de sus clases y métodos son selladas, por lo que de todas formas son inutilizables para programación normal.

Reglas de Ordenamiento Dinámicas

Este tipo de reglas permiten reordenar los resultados de una consulta por medio de un grupo de condiciones de forma similar a las Reglas de Consulta, pero que son aplicadas después de que el Motor de Búsqueda ya ha generado la lista de resultados. La parte encargada de

manejar la devolución de listas de resultados del Motor de Búsquedas utiliza palabras claves y XRANK para modificar el orden de la manera configurada.

Las Reglas de Ordenamiento Dinámicas se definen en el “Constructor de Consultas” de la WebPart de Resultados de Consulta (pestaña de “Ordenación”). En esta ventana, en el Combo-box de “Ordenar por” se debe seleccionar “Rank” para poder definir las Reglas de Ordenamiento Dinámicas. Luego, utilizando el vínculo de “Agregar regla de ordenación dinámica” se pueden definir nueve tipos de condiciones para “Promover” o “Disminuir” los resultados que cumplen la condición por un número determinado de niveles. Se pueden configurar múltiples condiciones por medio de un estamento “OR” (serán evaluados en orden de configuración).

Imagen 4.- Ordenación en la Web Part de Consulta de Búsqueda.

Las condiciones que se pueden utilizar son:

- El resultado contiene la palabra clave - Como su nombre lo indica, se puede promocionar o democoionar un resultado que contenga el termino configurado.
- El título contiene palabras clave - Igual a la anterior condición, pero que se aplica al título de la información solamente, no a su contenido, y el título puede contener más términos.
- El título coincide con las palabras clave - Igual a la anterior, pero el título debe ser igual al término configurado.
- La URL empieza por - La condición se aplica a todos los resultados en donde su ruta comienza con el termino configurado.
- La URL coincide exactamente - Permite manipular el orden de sitios específicos.
- El tipo de contenido es - La condición se basa en el Tipo de Contenido de la información, permitiendo promover todos los elementos creados con él.
- Coincidencias de extensión de archivo - Permite configurar la posición de archivos encontrados en los resultados basándose en su extensión (por el ejemplo documentos Word con extensión .docx).
- El resultado tiene la etiqueta - En donde la descripción o cualquier otro metadato coincide con el término configurado.

- Condición manual - Permite utilizar Regular Expressions para crear la condición.

Lo mismo que para las Reglas de Consulta Dinámicas, SharePoint no dispone de cmdlets de PowerShell para trabajar con Reglas de Ordenamiento, y las posibilidades de utilizar el Modelo de Objetos de Servidor son extremadamente reducidas, haciendo prácticamente imposible configurarlas de otra manera diferente a utilizando la Interface de Usuario.

Modelos de Ranking

El Motor de Búsqueda es responsable de asignarle un ranking a cada elemento que se encuentre en una consulta, basado en un número de factores definidos en el modelo de ordenación. Este modelo contiene las reglas y las operaciones matemáticas y lógicas que se deben aplicar a los resultados de la consulta. SharePoint Server 2013 dispone por defecto de catorce modelos de ranking que se aplican en diferentes contextos, tales como búsquedas en documentos o en información sobre personas.

Los modelos de ranking son archivos XML que se registran en SharePoint para que el Motor de Búsqueda los pueda utilizar. Aunque en SharePoint 2013 es posible modificar o crear nuevos modelos de ranking, Microsoft recomienda evitar hacerlo y utilizar las otras posibilidades descritas anteriormente para afectar el ranking. Muy probablemente esta es la razón por la que prácticamente no existe información disponible sobre la estructura y funcionamiento de estos modelos.

PowerShell para SharePoint 2013 dispone de cuatro cmdlets para trabajar con Modelos de Ranking:

- Get-SPEnterpriseSearchRankingModel - Enumera los modelos instalados y permite exportarlos a archivos físicos.
- New-SPEnterpriseSearchRankingModel - Despliega un nuevo modelo.
- Remove-SPEnterpriseSearchRankingModel - Elimina un modelo.
- Set-SPEnterpriseSearchRankingModel - Permite configurar las propiedades de un modelo.

Para exportar el modelo por defecto de SharePoint 2013 hacia un archivo físico utilice un script similar al siguiente:

```
$ssa = Get-SPEnterpriseSearchServiceApplication -Identity "Aplicación de servicio de búsqueda"

$myPropietario = Get-SPEnterpriseSearchOwner -Level ssa

$unModelo = Get-SPEnterpriseSearchRankingModel -SearchApplication $ssa -Owner $myPropietario

$unModelo.RankingModelXML > c:\unModelo.xml
```

Una vez exportado, un modelo puede ser modificado e importado de nuevo en SharePoint, preferiblemente con un nombre diferente para no alterar el modelo original. El siguiente script permite importar el modelo exportado en el paso anterior:

```
$myRankingModel = Get-Content c:\unModelo.xml

$myRankingModel = [String]$myRankingModel

$ssa = Get-SPEnterpriseSearchServiceApplication

$myPropietario = Get-SPEnterpriseSearchOwner -Level ssa

$newrm = New-SPEnterpriseSearchRankingModel -SearchApplication $ssa -Owner $myPropietario -RankingModelXML $myRankingModel
```

Los Modelos de Ranking consisten de diferentes "Características" ("Features") que, combinadas, forman el orden final de un elemento. El ranking se calcula para cada elemento en una consulta, y está basado en algoritmos matemáticos que pueden ser bastante complejos. El siguiente cuadro muestra las funciones de transformación para solamente las características de ranking estático y de proximidad (información proporcionada por Microsoft):

Transform	Formula	Parameters	Example
Rational	$y = \frac{x}{x+k}, \text{ where } x \geq 0$	k	<transform type="Rational" k="0.7"/>
Inverse Rational	$y = \frac{1}{1+kx}, \text{ where } x \geq 0$	k	<transform type="InvRational" k="0.7"/>
Linear	$y = \begin{cases} ax+b, & \text{where } 0 \leq x \leq x_{max} \\ a x_{max}+b, & \text{where } x > x_{max} \end{cases}$	a, b, x _{max}	<transform type="Linear" a="0.5" b="1.5" xmax="100.0"/>
Logarithmic	$y = \begin{cases} \log(x+b), & \text{where } 0 \leq x \leq x_{max} \\ \log(x_{max}+b), & \text{where } x > x_{max} \end{cases}$	b, x _{max}	<transform type="Logarithmic" b="2.4" xmax="1000.0"/>
Boolean	$y = \begin{cases} a, & \text{where } x \leq x_{max} \\ b, & \text{where } x > x_{max} \end{cases}$	a, b, x _{max}	<transform type="Boolean" a="2.5" b="1.2" xmax="12.5"/>
Datetime Boost	$y = \begin{cases} \text{ExpiredBoost}, & \text{where } x < 0 \\ \min(e^{ax-b}, y_{max}), & \text{where } 0 \leq x < x_{max} \\ \min(e^{ax_{max}-b}, y_{max}), & \text{where } x \geq x_{max} \end{cases}$	ExpiredBoost, a, b, x _{max} , y _{max}	<transform type="DatetimeBoost" a="2.5" b="1.2" xmax="12.5" ymax="5.2" expirboost="16.3"/>
Freshness	$y = \begin{cases} y_{future}, & \text{where } x < 0 \\ \frac{1}{1+cx}, & \text{where } x \geq 0 \end{cases}$	c, y _{future}	<transform type="Freshness" constant="0.033" futurevalue="2"/>

Imagen 5.- Modelos matemáticos para calcular el Ranking.

El modelo por defecto de ranking consta de siete de este tipo de características, que luego son agregadas por medio de modelos matemáticos que tratan de predecir la importancia del elemento en el conjunto de resultados. Los Modelos son altamente complejos de crear y testear; por ejemplo, el modelo por defecto que se exportó utilizando el script al principio de esta sección consta de 2.290 líneas de código XML.

Conclusiones

El Motor de Búsqueda de SharePoint 2013 calcula por defecto la relevancia de cada elemento en el conjunto de resultados de una consulta, permitiendo que los elementos que deben ser más interesantes para el usuario aparezcan al principio de la lista de resultados. Este ordenamiento se consigue por medio de modelos de Ranking que pueden ser creados y/o modificados. Pero existen otras formas para manipular el orden de los resultados, como por ejemplo por medio de reglas de ordenamiento que se pueden aplicar antes o después de que el Motor realice la búsqueda, por medio de la definición de Páginas Relevantes y la creación de Diccionarios de búsqueda.

GUSTAVO VELEZ

MVP de SharePoint Server

gustavo@gavd.net

http://www.gavd.net

SharePoint 2013 sobre Azure IAAS

Resumen

Azure es uno de los entornos en la nube que soportan el despliegue de una granja de SharePoint 2013. Como no todo siempre es tan fácil o simple en el mundo Microsoft, este artículo pretende contar nuestras experiencias con Azure y las máquinas virtuales de una granja de SharePoint 2013.

Artículo

Bien, bien, bien... como decirlo suavemente: SharePoint 2013 y Azure a priori no se llevan bien.

No obstante con un poquito de atención a los detalles es relativamente fácil montar una granja SharePoint 2013 sobre Azure, siempre desde el enfoque IAAS (Infraestructure As A Service).

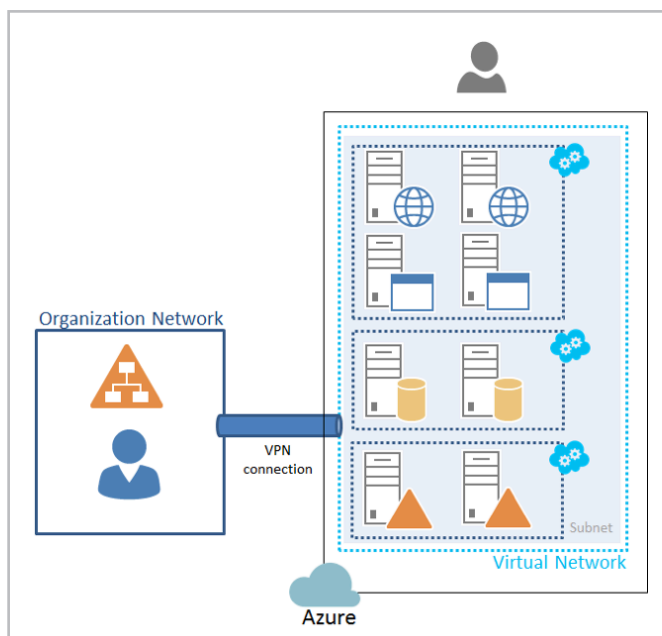


Imagen 1.- Diagrama de despliegue general. <http://msdn.microsoft.com/es-es/library/dn275958.aspx>

Para tener éxito en esta labor, es preciso tener presentes unos ciertos matices que, de momento, son limitaciones de Azure y que si no los tenemos en cuenta en la fase de planificación de la granja, nos pueden dar muchos quebraderos de cabeza.

No es el objetivo, ni mucho menos, de este artículo el explicar cómo configurar correctamente una granja SharePoint 2013 sino más bien, presentar una serie de recomendaciones a la hora de afrontar una instalación de una granja SharePoint 2013 distribuida sobre máquinas virtuales en Azure.

Las malditas IOPS vs SQL Server

Al afrontar una instalación de una granja SharePoint 2013 el primer hito es configurar un servidor SQL Server que proporcione un rendimiento adecuado. Los estándares normales de configuración OnPremise, donde la instalación por defecto suele funcionar bien no se aplican.

Es muy importante recordar, y la documentación de Azure se extiende en explicarlo, que cada virtual disk de Azure está limitado a 500 IOPS (Operaciones de Entrada / Salida por Segundo), por lo que es más importante que nunca aprovisionar al servidor de SQL Server con suficientes discos duros para poder distribuir nuestras bases de datos de servicio y contenido en función de su uso.

Esto es:

- Si todas las bases de datos están sobre un mismo virtual disk tenemos 500 IOPS.
- Si las BD de servicio están en un virtual disk y las de contenido en otro, tenemos 1000 IOPS.
- Si tenemos una BD de contenido con un uso intensivo y le damos un virtual disk para ella sola, esa BD tiene 500 IOPS a su disposición.

REPARTE y DISTRIBUYE!, Vale la pena.

Resumiendo, hay que saber evaluar cuantas BD vamos a tener, que uso tendrán y tener dos dedos de frente para distribuir las en diferentes discos virtuales, teniendo siempre en cuenta que dependiendo del tipo de máquina Azure que hayamos elegido para el Servidor SQL, tendremos la posibilidad de montar más o menos virtual disks. SharePoint, el invitado de piedra.

A SharePoint 2013, realmente, le da igual estar OnPremise que en Azure. La instalación y la configuración es la misma. Como recomendación personal, nunca usaría las plantillas

de Azure para el servidor/es de SQL ni para los servidores miembros de la granja SharePoint porque siempre me han ofrecido un rendimiento muy pobre. En su lugar suelo elegir la plantilla de Windows Server 2012 más actualizada y sobre ella instalo la versión de SQL Server 2012 o SharePoint 2013 que me interese. Poco más que añadir salvo ¡Cuidado con la cache!

¡Con la cache hemos topado!

Es vital recordar que los servidores que se montan en Azure vienen con el firewall cerradísimo. Aparte de los puertos estándar que se suelen abrir para SharePoint y/o SQL ahí que acordarse de abrir los puertos desde el 22233 al 22236 en todos los servidores SharePoint de la granja, porque seguro que no quieres problemas con el Distributed Cache Service.

¿Dónde estará mi token? (Singin)

Si tienes previsto montar varios frontales de una aplicación WEB de SharePoint en un ecosistema Azure, salvo que sea una aplicación extremadamente simple, olvídate del balanceador de carga de Azure (AKA Traffic Manager). Está reportado que fallan a la hora de transportar el token de sesión en el momento de balanceo.

Quieres montarlo, perfecto, balancea bien, aparentemente... pero espero que no te importe volver a introducir las credenciales si el Traffic Manager decide mover tu sesión a otro frontal... Por lo demás...

Para una aplicación con autenticación, no lo uses. Existen soluciones de terceros que se integran con Azure que solventan este problema, pero el balanceador nativo no es para el tipo de aplicaciones que se suelen desarrollar sobre SharePoint.

¡SQL HA OA, OMG! (¡Cuántas Siglas!)

En los tiempos que corren creo que estamos todos de acuerdo que la mejor solución para montar una alta disponibilidad del backend de SQL para SharePoint es Always On, si estamos OnPremise.

¿Tenemos el SQL en Azure? Mal... A día de hoy puedes montar un clúster HA Always On (High Availability con Always On) de SQL en Azure, con la salvedad que no puedes asignarle una IP al servicio del clúster (ya avise de que íbamos mal), pero este hándicap se puede salvar de dos formas que yo conozca, pero ninguna es buena al 100%

Las soluciones que conozco y propongo son:

- Montas los servidores SQL en un grupo de disponibilidad diferente al de los Servidores de la granja SharePoint y configuras un EndPoint en el grupo de disponibilidad de SQL que redirija al servidor SQL principal y al secundario como failover.

Esta solución parece buena, salvo por el pequeño matiz de que estas ofreciendo el clúster SQL con una IP pública disponible desde todo internet.

- Configuras el clúster AlwaysOn en el mismo grupo de disponibilidad que la granja SharePoint, pero como no le puedes registrar una IP y un nombre como punto de entrada al clúster en Azure tienes que engañar a SharePoint.

El workaround consiste en configurar en SharePoint todas las bases de datos para que usen como Primary Database Server el nodo principal de SQL y como Secondary Database Server el nodo secundario, como si en SQL hubiéramos configurado un Mirror o un LogShipping. En la práctica funciona y realmente la respuesta ante fallos que proporciona el tener un servicio de clúster monitorizando los SQL y poniendo on-line las BD de failover frente a un mirror o un log Shipping vale la pena el esfuerzo.

Esta última solución es la que suelo implementar y la que recomiendo, me parece más segura desde el punto en el que no está accesible desde internet y además es fácil configurar SharePoint modificando la propiedad 'FailOver Database Server' de todas las bases de datos desde un script PowerShell.

Entonces, ¿me subo a Azure?

Sí, claro. Azure es una gran solución para desplegar soluciones IAAS multiserver complejas y olvidarte de la gestión de tu CPD. No es la panacea y su arquitectura ofrece retos que los profesionales de IT debemos afrontar y superar, pero las contraprestaciones que ofrece en muchos casos se ven superadas por los costes y esfuerzos de mantener las infraestructuras OnPremise, sumándole que la escalabilidad frente a picos de sobrecarga es muy superior a lo que cualquiera podemos y estamos dispuestos a afrontar en nuestros servidores locales.

En conclusión, Azure es una herramienta más a disposición de los profesionales de IT y conociéndola y midiendo adecuadamente las expectativas, Azure se puede expresar y obtener grandes resultados; muchas veces superiores a los esperados de las 'más robustas' instalaciones OnPremise, que cuando las analizamos y comparamos con Azure, se ruborizan por sus carencias.

LUIS EMILIO LÓPEZ

SharePoint IT Manager en Encamina

llopez@encamina.com

<http://www.encamina.com>

SharePoint 2013: ¿Solución o Aplicación?

Resumen

El nuevo modelo de Aplicaciones de SharePoint 2013 implica que a la hora de desarrollar soluciones sobre la plataforma tengamos que determinar en primer lugar si se va a crear una solución clásica o bien una Aplicación para dar respuesta a los requerimientos de negocio planteados.

Artículo

SharePoint es una plataforma empresarial con diversas funcionalidades como puede ser Gestión Documental, Social Enterprise, Comunidades y sitios, Gestión de Contenido, Búsquedas, Inteligencia de negocio, desarrollo de soluciones, aplicaciones, flujos de trabajo, etc.

Los clientes suelen empezar por implementar su intranet con SharePoint, con sus sitios departamentales y demás, aprovechando las funcionalidades que tiene el producto para cambiar la forma de trabajar de la compañía e introducir el concepto de Colaboración en ella.

Por el camino de todo este proceso, que se puede implementar de una forma sencilla, empiezan a aparecer las necesidades de incluir en los procesos las reglas de negocio de la compañía, que podemos hacerlo mediante flujos de trabajo o extendiendo la plataforma implementando lo que llamamos Soluciones de SharePoint.

Hasta aquí, nuestras herramientas de desarrollo pasan por el navegador y las funcionalidades que SharePoint tiene out-of-the-box para personalizar la plataforma que permiten hacer casi un 70-80% de lo que las empresas necesitan para su Intranet, a tener que implementar flujos de trabajo, metadatos empresariales, tipos de contenido, WebParts con funcionalidades específicas y demás artefactos que SharePoint permite implementar mediante Visual Studio.

Lo normal es que implementemos ese conjunto de funcionalidades extras utilizando una Solución de SharePoint, empaquetada desde Visual Studio, y os pongo un ejemplo:

Un cliente quiere una gestión documental, que sea transversal a toda la compañía, con flujos de trabajo

asociados a tipos de documentos específicos y con la automatización de algunos campos de los metadatos de los documentos.

Nos podríamos plantear, casi sin ningún problema, implementar estas funcionalidades con lo que en SharePoint se llama un No-Code Solution que implicaría crear los tipos de contenido (algo similar a un tipo de documento), metadatos y flujos de trabajo con el navegador o con SharePoint Designer. La desventaja de todo esto que es perdemos el foco de que todo desarrollo conlleva cambios y que con este tipo de soluciones no podemos poner los controles adecuados para tener un control de versiones de esos cambios o poder realizar un conjunto de pruebas adecuadas.

Lo habitual es implementar esta necesidad usando Visual Studio de la siguiente forma:

- Una característica de SharePoint que cree las columnas y los tipos de contenido. Hay que tener en cuenta que cuando hayan cambios en las columnas o en los tipos de contenido, hay que realizar las oportunas actualizaciones del esquema de SharePoint.
- Para automatizar los valores de las columnas, JSLink (renderizado en cliente) en las columnas o, lo que me parece la solución más elegante, crear campos personalizados con la lógica de negocio necesaria para el automatismo. Por ejemplo, necesitamos que un campo consulte un servicio web y se rellene con el número de registro que le corresponda, o, necesitamos que un campo se rellena con el valor del campo departamento al que pertenece el usuario.
- Los flujos de trabajo se implementan con Visual Studio, asociados a un tipo de contenido. La idea es que se use donde se use ese tipo de contenido, tenga asociado el proceso de negocio que le corresponda.
- Si necesitamos personalizar las vista de los documentos, podemos incluir JSLink que nos permita controlar desde el lado de cliente como se renderiza la información.

Esto al final se convierte en un paquete de solución de SharePoint o wsp, con el que podemos desplegar para

que se creen automáticamente todo este contenido y ponerlo disponible para el usuario. Por su puesto, con un control de versiones y, si fuera necesario, entorno de Pre-Producción, integración continua, etc. Todo esto respaldado por una API gigantesca con diversos sabores:

- Modelo de Objetos de Servidor, con todas las funcionalidades que SharePoint nos ofrece.
- Modelo de Objetos de Cliente .NET, limitadas las funcionalidades por seguridad.
- Modelo de Objetos de Cliente JavaScript, limitadas las funcionalidades por seguridad.
- API REST/OData, limitadas las funcionalidades por seguridad.
- Modelo de Objetos de Cliente Silverlight y Windows Phone, limitadas las funcionalidades por seguridad.

Esta API nos permite extender la plataforma e incluir lógica de negocio dentro de los procesos que se ejecutan en SharePoint. Pero ¿qué pasa cuando necesitamos almacenar grandes cantidades de filas en una lista o lógica de negocio que conlleva un almacenamiento de la información relacional?

Empezamos con las dudas y la primera es: SharePoint NO es una base de datos relacional aunque se puede simular. Tened claro que el almacenamiento de SharePoint, aunque al final sea SQL Server, no es el que nos permite

tener un modelo de datos relacional, como mucho, lo podemos encajar en un tipo de almacenamiento NoSQL. La siguiente es que las listas y bibliotecas de documentos de SharePoint tienen unos límites que Microsoft no recomienda sobrepasar. Hablamos de unos 5.000 elementos que se pueden mostrar por vista de lista y unos 50.000 permisos exclusivos en ellas. No hay problema, los usuarios dicen que no van a tener más de 5 registros por mes y que estamos sobrados con esos límites. JA JA JA Los usuarios siempre mienten!

¿Y por qué os cuento esto? En SharePoint 2013 nació un nuevo modelo de desarrollo llamado Aplicaciones de SharePoint, del que he tenido el placer de escribir un libro (http://www.campusmvp.es/catalogo/Product-Todo-sobre-Aplicaciones-para-SharePoint,-Office-y-Office-365_181.aspx) junto con algunos amigos. Modelo en el que creemos y parece ser el futuro del modelo de desarrollo para SharePoint. Modelo que tenemos que empezar a aprender y aplicar a nuestros desarrollos, en la medida de lo posible.

Microsoft recomienda hacer todo con Aplicaciones si tenemos SharePoint 2013 y pienso que ni tanto ni tampoco. En estos momentos, el modelo de Aplicaciones no permite cumplir con todos los requisitos que los usuarios piden a SharePoint, pero si con muchos que, por suerte o por desgracia, ya estamos usando y están generando quebraderos de cabeza por problemas de rendimiento y mantenibilidad.

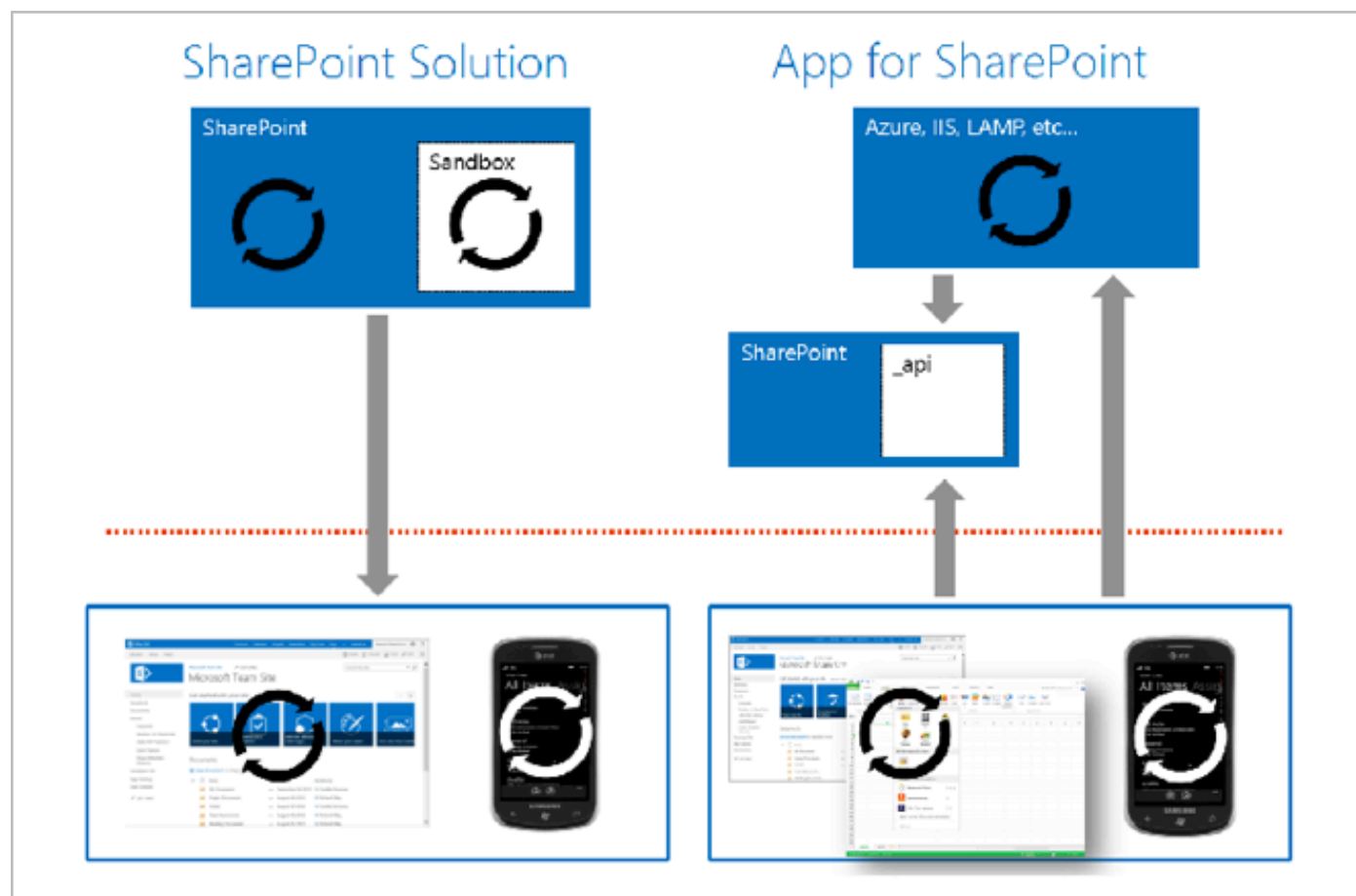


Imagen 1. Entorno de ejecución de Soluciones y Aplicaciones. [http://msdn.microsoft.com/es-es/library/office/dn268593\(v=office.15\).aspx](http://msdn.microsoft.com/es-es/library/office/dn268593(v=office.15).aspx)

Siendo fríos, calculadores y pensando en que este modelo es el futuro, habría que tener en cuenta los siguientes casos para elegir uno u otro:

- En SharePoint Online, prácticamente tenemos que ir al modelo de Aplicaciones. Aunque podemos implementar Soluciones Sandbox o inyectar JavaScript, nos estamos limitando bastante en el ámbito de aplicación de la lógica de negocio y las constantes actualizaciones de Office 365 no ofrecen garantía de poder modificar el DOM con JavaScript y que funcione en nuevas versiones, como ya lo estamos sufriendo. Adicionalmente, no es recomendable desplegar soluciones Sandbox con código que Microsoft las considera como “deprecadas”.
- Si hablamos de gestión documental que necesita lógica de negocio, para establecer procesos de ECM o soluciones Records Management, esto tiene que ser una Solución si o sí. Entre las bondades de SharePoint se encuentra la gestión documental, los tipos de contenidos, la gestión de registros, y todo lo que tenga que ver con el Enterprise Content Management (<http://blogs.encamina.com/negocios-sharepoint/2014/04/14/el-valor-de-una-plataforma-de-gestion-de-contenido-empresarial/>). Con el modelo de aplicaciones tendríamos que desarrollar funcionalidades que SharePoint ya tiene y que podemos extender con una Solución de SharePoint bien implementada.
- Para aplicaciones con modelos de datos relacionales o grandes cantidades de filas no pretendas usar listas porque tarde o temprano te saltarás los límites, gana en salud desarrollando una Aplicación.
- Una organización madura en SharePoint empieza a implementar aplicaciones LOB (Line of Business) o soluciones departamentales en la Intranet, y en un 90% deberíamos de elegir una Aplicación pensando en ASP.NET MVC, LightSwitch, Access o PHP. Da igual la tecnología y esa es la clave de las aplicaciones para que todo forme parte de nuestra Intranet. Para el otro 10% solemos desarrollar pequeños WebParts con consultas a datos externos o Business Connectivity Services.

¿Pero qué pasa con los equipos de desarrollo?

Buena pregunta porque pienso que los equipos de desarrollo de SharePoint no están preparados para dar el salto al modelo de Aplicaciones. Esto que parece una desventaja, puede ser una gran ventaja y os explico el por qué.

Una Aplicación es, básicamente, una web con cierta funcionalidad implementada a nivel de servidor o cliente, sin importar la tecnología que se use en cada caso. Eso quiere decir, que un desarrollador de SharePoint que lleva desde la versión 2007 implementando WebPart,

Una Aplicación es, básicamente, una web con cierta funcionalidad implementada a nivel de servidor o cliente, sin importar la tecnología que se use en cada caso.

Features, Event Receiver, Workflows y demás artefactos de SharePoint, no tiene los conocimientos adecuados para el desarrollo con HTML5, JavaScript, ASP.NET MVC, que son tecnologías actuales de desarrollo. Pero si juntamos a estos buenos desarrolladores de SharePoint con buenos desarrolladores de tecnologías actuales, a los que no les tenemos que enseñar SharePoint, conseguimos el equipo ideal.

El modelo de Aplicaciones de SharePoint nos permite abstraer de la plataforma a desarrolladores de Front-End que, por ejemplo, están desarrollando una SPA (Simple Page Application) con AngularJS y, con una buena organización del proyecto, permitir que un buen desarrollador de SharePoint ponga todo su conocimiento en aquellos puntos en los que es necesario integrar la Aplicación con la plataforma.

Con esto no quiero decir que los nuevos desarrolladores de SharePoint no tengan que conocer el producto, en realidad quiero decir que el conocimiento avanzado que es necesario para hacer una buena solución de SharePoint y que se consigue con muchos años de experiencia y dolor con el producto no es tan importante para avanzar en la mayoría del código de la aplicación. Como toda evolución natural, un desarrollador de SharePoint tendrá que conocer de HTML5, JavaScript o la tecnología que se use para el desarrollo de las aplicaciones, además del conocimiento que se presupone de SharePoint. Vamos, lo que en estos momentos consideramos un Super-Developer. Mientras tanto, tenemos que ser capaces de juntar estos dos perfiles y que cada uno aprenda lo mejor o necesario del otro para caminar por este nuevo camino.

ALBERTO DIAZ MARTIN

MVP SharePoint

adiazcan@hotmail.com

[@adiazcan](https://twitter.com/adiazcan)

<http://geeks.ms/blogs/adiazmartin>

Entrevista a Andrés Rojas

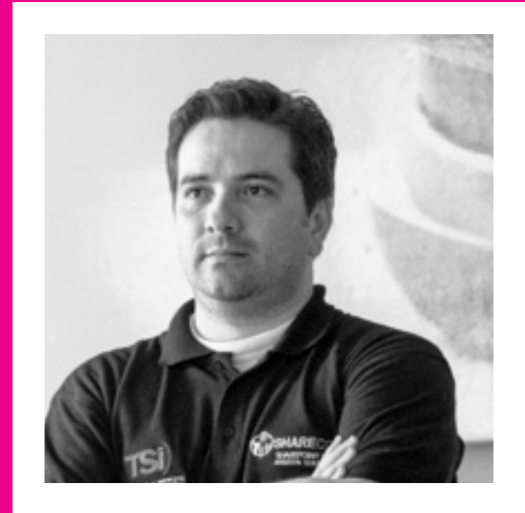
Mi nombre es Andrés Felipe Rojas Parra y nací en 1976 en Florencia, Caquetá, Colombia. Actualmente vivo en la ciudad de Bogotá, Colombia.

Desde hace aproximadamente 10 años me dedico a tecnologías Microsoft y unos 8 años a SharePoint y Project Server en todas sus versiones.

SharePoint para mí lo ha sido todo, soy lo que soy hoy en día gracias a esta fenomenal herramienta de colaboración.

He participado de proyectos magníficos aquí en Colombia con empresas privadas y del gobierno, así como, en algunas ocasiones, he colaborado directamente con Microsoft en casos de soporte premier. Desde que empecé a trabajar con SharePoint he tenido la oportunidad de crecer interactuar con grandes personalidades a nivel mundial como es el mundo de los MVPs. Hace dos años recibí este galardón, como MVP en SharePoint, por cuenta de Microsoft y cuyo recibimiento, me ha impulsado a crear mi propia comunidad de conocimiento de SharePoint, ShareCol (<http://www.sharecol.com>).

Actualmente trabajo para Pacific Rubiales Energy, compañía petrolera en Colombia, donde uno de mis principales roles es el de mantener todas las aplicaciones de negocio que



corren sobre aplicaciones Microsoft como SharePoint, Project Server, CRM Dynamics entre otras.

Hace ya algunos años escribo en mi blog <http://arojaspa.blogspot.com> y en el blog de la comunidad SharePoint de Colombia <http://www.sharecol.com/blogs>, ahí podrán encontrar diversos post de desarrollo, IT y de SharePoint en General. Para aquellos que siguen twitter, mi cuenta es @arojaspa.

Por qué y cómo empezaste en el mundo de la tecnología?

Empecé en este mundo de la tecnología gracias a mi papá que, desde que estudiaba en el colegio, siempre llegaba todos los viernes a la casa con una revista de Ingeniería Electrónica. Era fantástico saber que el viernes iba a tener esa revista ya que siempre uno de mis sueños fue trabajar para la NASA y pensaba que aprendiendo de Electrónica lo iba a lograr. Lamentablemente en la época de colegio, carecíamos de algunos recursos tecnológicos en la ciudad donde vivía.

Cuando llego a la universidad, a estudiar Ingeniería Electrónica es donde me enfoco en el área de desarrollo de aplicaciones con Visual Basic, sí, ese mismo, el de la pantalla azul. Desarrollé varios aplicativos para proyectos de control robótico y posicionamiento global, también participé en un concurso con la MIT (Massachusetts

Institute of technology) donde me gané media beca para estudiar astrofísica, lamentablemente no tuve la oportunidad de gozar de dicha beca. Eso me motivó a continuar enfocando mis esfuerzos en hacer desarrollos de software para controlar Robots con inteligencia artificial. Me gradué como Ingeniero electrónico y procuro mantener la sinergia entre desarrollar aplicaciones enfocadas a controlar requerimientos de negocio.

¿Cuáles son tus principales actividades tecnológicas hoy en día?

Actualmente trabajo como Business Technical Application Leader en Pacific Rubiales, donde tengo a mi cargo toda la plataforma de Colaboración de la organización que funciona en una granja de SharePoint de aproximadamente 8 servidores. Adicionalmente, también tengo a mi cargo

varias aplicaciones que han sido desarrolladas para CRM Dynamics que son críticas para el negocio. Mis roles son más bien compartidos entre IT Pro y Developer ya que, debo validar los desarrollos de software que realizan nuestros proveedores tanto para SharePoint como para CRM. En paralelo, como director de la Comunidad SharePoint de Colombia, estamos en la organización del SharePoint Saturday Bogotá (<http://www.spsbogota.co>). Tratamos de mantener una constante de transmisión de conocimiento de SharePoint 2010 y 2013 con los grupos de estudio de la comunidad realizando webcast y cursos presenciales enfocados a usuario final.

¿Cuáles son tus principales actividades NO tecnológicas hoy en día?

Soy padre de dos hermosos hijos, Juan Andres de 12 años y Mariana de solo 15 meses. He sido un bendecido por Dios debido a que mis dos hijos nacieron exactamente el mismo día 12 de enero, solo que con una diferencia de 11 años y cuatro (4) horas nada más. Mantengo una relación espectacular con la mejor de las mujeres del mundo, Jenny Martinez.

Hay que balancear lo profesional con lo personal, ni más para uno, ni menos para el otro. Ese equilibrio es vital para poder mantener claro tu horizonte.

Procuró compartir los fines de semana completamente a ellos y a mis padres (Ivan Rojas y Clara Parra) que viven en una población cercana a Bogotá, como a dos horas aproximadamente. Viajo constantemente con mi familia a visitarlos con mis hermanos (Ángela y Juan Manuel). Tengo otro hermano (Oscar Iván) que lamentablemente no vive en Colombia por cuestiones de trabajo, pero procuro hablar con él cada vez que el corazón lo siente así. Se les quiere mucho.

¿Cuáles son tus hobbies?

No soy de muchos hobbies, practico el Golf. Me encanta ver películas de acción, de ciencia ficción y películas de tipo motivadoras como The Blind Side, We are Marshall, Invictus, Invencible entre otras.

Soy el fan número uno de la Guerra de las Galaxias, creo que me he visto la saga completa unas 20 veces, además, ese gusto lo ha sacado mi hijo mayor y es bien interesante cuando nos sentamos a ver las películas, es un rato súper espectacular compartirlo con él. Creo que Juan Andrés será el fan número dos de la saga.

¿Cuál es tu visión de futuro en la tecnología de acá a los próximos años?

La ola de la tecnología nos ataca cada vez con más rapidez que uno no se da cuenta a qué horas todo cambió. Soy creyente en que en algún momento vamos a depender cada vez más de ella. La nube, las nuevas tecnologías de los dispositivos móviles, el crecimiento en la detección de órganos biométricos hacen que se deba tener claridad hacia dónde vamos y que vamos a hacer con esa información. Es importante tener el control de las cosas, quizás con el apoyo de estas grandes tecnologías de almacenamiento, de comunicación y demás, podríamos sentir que la seguridad de nuestra información puede llegar a ser vulnerada. Somos lo que somos porque lo somos, pero: ¿Qué pasaría si alguien más quiere ser lo que uno es? Con el avance hoy en día es un factor clave del que se debe tener mucha certeza de cómo mantener nuestra información segura.

Dentro de unos años, vamos a tener la posibilidad de mantener una conversación con personas que no hablen nuestro idioma simplemente teniendo un chip traductor implantado en nuestra cabeza, vamos a ver asentamientos de personas en la luna y en Marte. Si allá vamos, ¿Creen que tenemos límites en nuestra imaginación para hacer lo que queremos hacer? Hay un dicho que me gusta mucho y esto es propiedad intelectual de mi madre “Ténganle miedo a lo que quieren hacer, porque lo hacen...”. El límite lo pones tú.

ANDRÉS FELIPE ROJAS PARRA

MVP SharePoint

Director de la Comunidad SharePoint de Colombia

andres.rojas@geeks.ms

[@arojaspa](#)

<http://arojaspa.blogspot.com>

SharePoint 2010 Deferred site collection upgrade with custom SharePoint solutions

Summary

In this article we will review the deferred site collection upgrade technique available in SharePoint 2013 as mechanism to ensure that custom Solutions working in a SharePoint 2010 environment will continue working in SharePoint 2013.

Article

We have been struggling with this concept for a while, being an ISV with many customers who use our products on SharePoint 2007/2010 and are interested in upgrading to SharePoint 2013, but using the deferred site collection upgrade approach. You see, in this approach you upgrade to a SharePoint 2013 farm but still run your 2010 sites in SharePoint 2010 user interface, while allowing site owners to gradually upgrade to 2013 interface gradually.

The challenge this poses for us was simple: which version of our solution should a customer install on such a hybrid farm?

A SharePoint 2010 solution will work on the 2010 experience sites, but will not work on those that were already upgraded to 2013 or on new sites created in 2013 experience, while a SharePoint 2013 solution will work on the upgraded sites or new sites, but not on sites still using the 2010 user interface.

Installing both versions seemed to be the solution, but we had a few issues with that approach:

1. Our 2010 and 2013 WSP solutions have the same solution ID, thus cannot be installed on the same farm at the same time.
2. Both 2010 and 2013 WSP packages contain the same DLLs assemblies, and so if we retract one – it will remove the shared DLLs and break the other. So, when upgrade is done and we want to remove the 2010 package it will break the 2013 package. After working closely with the good people at Microsoft, we ran some tests and came up with the following procedure that would work for us, and possible for everyone in this situation. Assuming you have a solution which includes a 2010 version called “MySolution2010.wsp” and a 2013 version called “MySolution2013.wsp”:

1. Install the 2013 solution on your 2013 farm (which is supposed to support 2010 backward compatibility sites).
2. Create a new package (called “MySolution2010BC.wsp”) that installs only the needed resources and install it on your farm. This

package will be used during the deferred site collection upgrade period.

3. Once that period is over and everything was upgraded – remove the “MySolution2010BC.wsp” package.

This approach seems to be working great on our testing, but we still had the 2 issues to deal with (DLLs and solution ID) and I didn’t want to burden our R&D with maintaining and producing another package for every product.

So, I am happy to share with you a new codeplex project we’ve just released for this purpose: “WSP Repackage tool” found here: <https://wsprp.codeplex.com/>

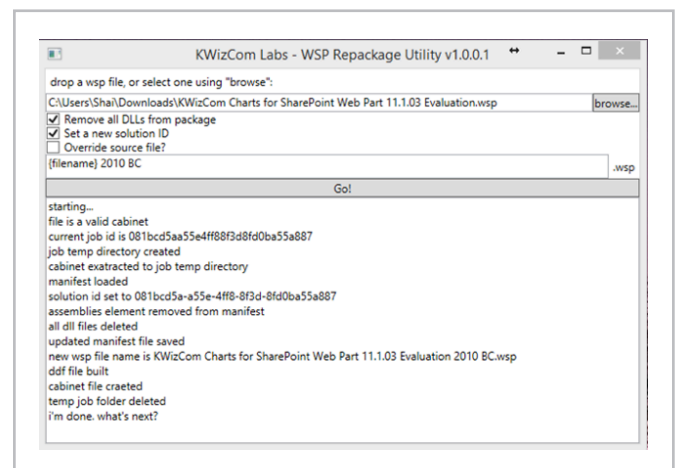


Image 1.- WSP Repackage tool.

This tool will take your existing 2010 solution package, and repackage it as a new WSP file with a new solution ID and without the assemblies inside. This new package will be the “2010 BC” package that you can safely install on your SharePoint 2013 farm along with the SharePoint 2013 solution to be able to support both 2013 native sites and upgraded sites that still use 2010 user interface.

If you have thoughts, comments please feel free comment on the discussion in the Codeplex project <https://wsprp.codeplex.com/>.

SHAI PETEL

SharePoint Server MVP

VP R & D MCPD, MCT, MVP

shai@kwizcom.com

[Blog / Twitter](#)

Novedades en SQL Server 2014

Resumen

En este artículo veremos las principales novedades que vienen en la última versión de SQL Server liberada por Microsoft: SQL Server 2014.

Artículo

Como todos sabemos SQL Server es una de las soluciones de plataforma de datos que más ha evolucionado en los últimos años y versiones, y esta vez para su nueva versión 2014 no iba a ser menos. En las próximas líneas intentaré daros unas pinceladas sobre las novedades que encontraremos en esta versión.

Lo incambiado

Para sorpresa de los que esperan grandes cambios de versión a versión, SQL Server 2014 no ofrece grandes novedades para Integration Services, Analysis Services, Replicación y la única novedad para Reporting Services es el soporte para navegadores Chrome. Desde mi punto de vista esto se debe a que estamos ante una versión “de salto”, donde el gran cambio viene en lo que podemos considerar el core del producto (OLTP) y donde estamos viendo la tendencia hacia entornos cloud y Apps.

In-Memory OLTP (Hekaton)

Para gustos los colores dicen, pero este es de lejos uno de los cambios más importantes de esta versión. El motor OLTP en memoria reduce de forma drástica los tiempos de I/O requeridos por las aplicaciones y según información oficial hablamos de hasta una mejora de velocidad de 20x. Otra de las ventajas es que podemos tener en una misma base de datos el modelo híbrido, es decir, tablas en memoria y tablas en disco pudiendo trabajar con ellas al mismo tiempo en una misma consulta, lo que nos permite jugar con los criterios de coste de la velocidad que necesitemos en cada caso. Para ayudar a visualizar la forma en la que podríamos migrar nuestras consultas y procedimientos almacenados contamos con un asistente que validará nuestro modelo actual y nos dará información sobre las características de la migración.

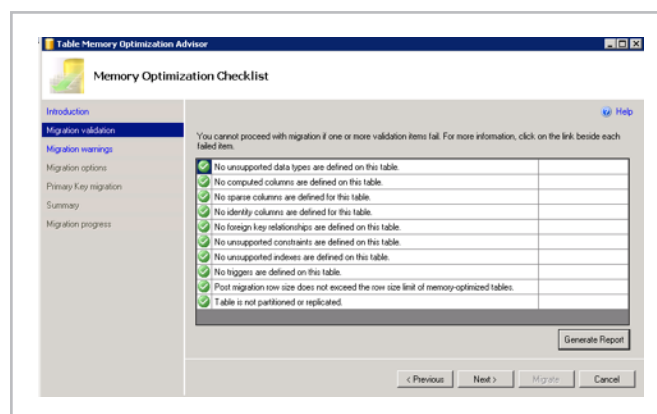


Imagen 1.- Checklist de optimización de memoria.

Azure

SQL Server 2014 ofrece importantes novedades con respecto a Azure. En primer lugar nos permite integrar modelos AlwaysOn con Azure, lo que implica tener un modelo mixto de recuperación ante desastres que nos permitirá trabajar en grupos de máquinas virtuales Azure ante un fallo de nuestras arquitecturas de datos locales, y para mejor todo gestionado desde del mismo SQL Server Management Studio. La segunda novedad es que nos permite realizar nuestros backups directamente en Azure de forma automática, cosa que siempre echamos en falta en ese preciso instante en que necesitamos un respaldo. Por último, otra de las novedades es la posibilidad de contar con los ficheros de datos de nuestra base de datos en Azure. Es decir, podemos tener nuestra base de datos corriendo de forma local pero con los ficheros que la soportan almacenados en Azure (en contenedores BLOB). Aunque a simple vista se desprenden las ventajas de este modelo en lo que respecta a migración y alta disponibilidad, debemos tener mucho cuidado con el rendimiento que implica, ya que nuestras transacciones están enviando y recibiendo todos sus datos asociados a través de internet.

SQL Server 2014 ofrece importantes novedades con respecto a Azure.

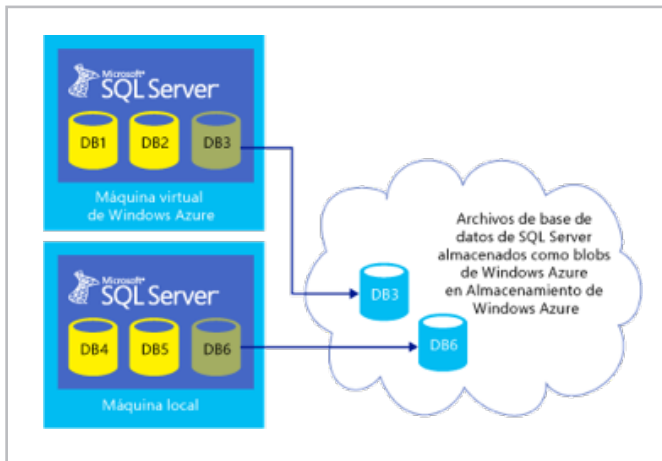


Imagen 2.- Arquitectura de almacenamiento de archivo de bases de datos en BLOBs de Azure.

ColumnStore

Los que trabajamos con modelos datawarehouse nos alegramos cuando se introdujo este concepto en SQL Server 2012 gracias a la gran velocidad que otorgaba para algunas consultas, pero para decepción de algunos nos encontramos con la limitación de que la tabla involucrada en este tipo de consultas debía ser de solo-lectura. Esta versión ha eliminado la limitación, lo que nos permite realizar cambios en la tabla en cuestión sin necesidad de borrar y volver a crear el índice ColumnStore.

Duración de las Transacciones

Como todos sabemos, uno de los problemas al utilizar transacciones es la necesidad de “esperar” que la transacción termine para devolver el control a la aplicación y esto implica que debemos esperar a que el LOG termine de registrar la transacción. En SQL Server 2014 tenemos un nuevo tipo de transacción delegada o retrasada, lo que implica que se devuelve el control antes de que la transacción sea registrada en el log y puede ser controlada a nivel de COMMIT, ATOMIC y obviamente a nivel de base de datos. Hay que tener muy en cuenta que modelo utilizar en cada caso, ya que estamos sacrificando seguridad en pro del rendimiento. Con esto quiero decir que hasta ahora debíamos esperar a que las transacciones terminaran para volver a tener el control, lo que nos garantizaba la seguridad en la información que se está leyendo pero nos penalizaba en entornos en los que teníamos un acceso a disco más tardío o cuellos de botella a nivel de log. Con la posibilidad de no esperar por la escritura del log ganamos esta velocidad, pero a cambio debemos estar dispuestos a pérdida de información o lectura “sucía”.

Business Intelligence

Por último no podía dejar este artículo sin mencionar los cambios en esta materia. Aunque las novedades no son específicas de SQL Server 2014, las mejoras de Office 365 y PowerBI implican una gran apuesta por estas tecnologías y en la que el usuario ha ganado gran relevancia en esta edición. En primer lugar PowerPivot ha dejado de ser un add-in y ya forma parte de Excel 2013 lo que nos ofrece un potente motor de tratamiento de información para el usuario final. Power View ha ganado varios tipos de nuevos reportes y el soporte para DAX (Data Analysis Expressions) es una mejora sustancial. Por último dos nuevos actores como PowerQuery y PowerMap no dejan al usuario indiferente. PowerQuery permite al usuario extraer información de toda clase de orígenes de datos no normalizados (Azure, ficheros de texto y XML, oData, Hadoop, etc) e integrarla dentro del modelo de información de forma sencilla e intuitiva sin tener idea de lo que es un modelo de datos o un modelo relacional, con todo lo bueno y lo malo que esto implica. PowerMap es un modelo de visualización 3D geográfico que permite posicionar la información sobre mapas Bing con una presentación realmente espectacular. Todos estos actores pueden combinarse en Office 365 en portales específicos con el objetivo de hacer que esta información esté siempre disponible y en donde se introducen novedades como cuadros de mando interactivos, presentaciones de datos y búsqueda de indicadores entre otras.

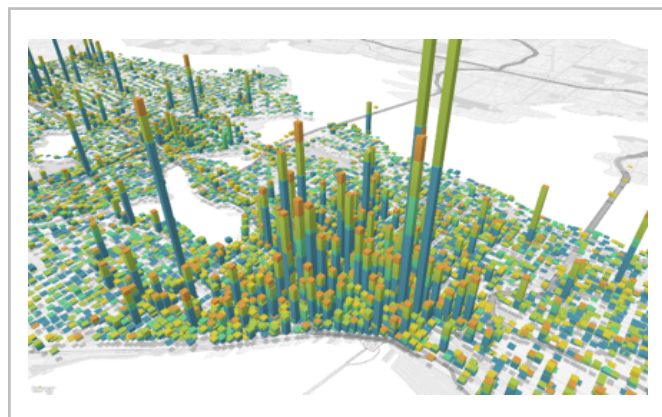


Imagen 3.- Ejemplo de PowerMap.

JAVIER MENÉNDEZ PALLO

Director Oficina Centro – España y Portugal

ENCAMINA

jmenendez@encamina.com

Novedades en el manejo de Bibliotecas de Documentos en SharePoint Online

Resumen

Desde el lanzamiento público a nivel global de Office 365 hace ya más de cuatro años, Microsoft siempre ha sido fiel a su promesa de ir añadiendo mejoras e innovaciones continuas a su plataforma de productividad en la nube: SharePoint Online. Al mismo tiempo, las nuevas capacidades añadidas están pensadas para que los usuarios las puedan usar desde cualquier dispositivo ya sea un equipo de sobremesa, un portátil, una tableta o un Smartphone.

Artículo

Los flujos de trabajo ("Workflows") son una herramienta indispensable para diferentes necesidades colaborativas que se desarrollan en la plataforma Microsoft SharePoint. La necesidad de automatizar procesos donde se relacionen adecuadamente eventos, personas, recursos y tareas para lograr un objetivo, hacen necesario que esta plataforma esté configurada y administrada con un alto rendimiento para alcanzar las miles de peticiones que realizan las soluciones colaborativas allí implementadas.

SharePoint es una plataforma extensa, con diversas funcionalidades y complejos componentes que requieren estar bien administrados por un completo y estructurado plan de gobierno. Dicho plan debe ayudar a mantener una estructura organizada, un crecimiento controlado y un alto desempeño de las prestaciones según la topología seleccionada. De esta forma se podrá evitar un desbordamiento inesperado sin caer en alguna de sus limitaciones.

Novedades en la gestión y uso Bibliotecas de Documentos en SharePoint Online

Desde hace unas semanas, cuando un usuario accede a sus sitios de SharePoint Online "se puede" encontrar (Nota: Los cambios que se detallan en este artículo no están siendo aplicados de forma global por Microsoft a todos los tenant de SharePoint Online en Office 365, sino que la aplicación se está realizando de forma incremental por lo que es posible que no todo usuario de SharePoint Online tenga disponibles estas capacidades) con que las bibliotecas de documentos del sitio cuenta con una serie de acciones que facilitan realizar tareas comunes sin necesidad de pasar por la Cinta de la biblioteca y muy enfocado a que las tareas que habilitan estas acciones se puedan realizar de forma sencilla

desde un dispositivo tipo tableta o Smartphone.

Las opciones disponibles varían en función de si se está accediendo directamente a la biblioteca o de forma indirecta a través de una vista en una WebPart:

- Para el caso en el que se acceda al contenido de la biblioteca a través de la correspondiente WebPart (Nota: Cada lista / biblioteca de documentos de un sitio de SharePoint cuenta con una WebPart de tipo XsltListViewWebPart que puede ser añadida en cualquier página del sitio para facilitar la visualización y manejo de la información / documentos contenidos en la lista / biblioteca), las nuevas opciones disponibles son: "cargar", "modificar" y "compartir".

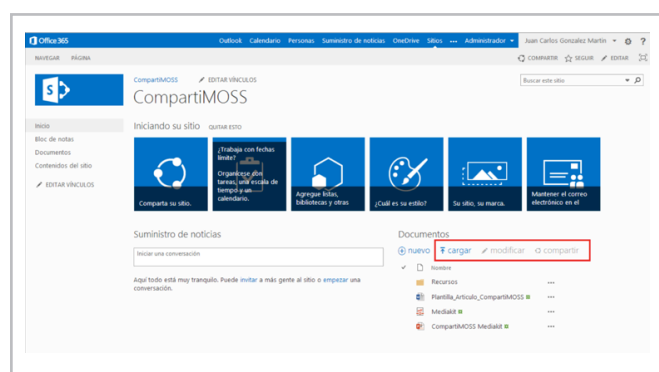


Imagen 1.- Nuevas opciones disponibles en una vista de una biblioteca de documentos.

- En el caso de que se acceda directamente a la biblioteca de documentos, a las acciones anteriores se añaden las siguientes: "Sincronizar" y "Administrar".

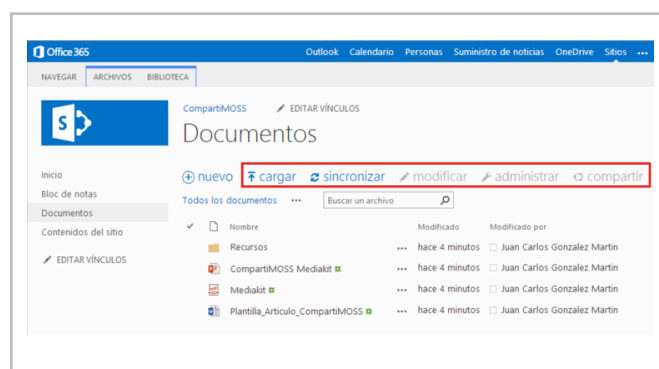


Imagen 2.- Opciones disponibles cuando se accede a la biblioteca de documentos.

Si analizamos las opciones en detalle y cuando se pueden utilizar, al igual que sucede con la Cinta de la biblioteca nos encontramos con dos tipos de acciones

- Acciones de biblioteca, que se engloban en las acciones “cargar” y “sincronizar”. La acción “cargar” permite cargar un nuevo documento en la biblioteca de documentos a través del clásico cuadro de diálogo de carga de documentos de SharePoint. La acción “sincronizar” permite sincronizar de forma local los documentos almacenados en la biblioteca utilizando el cliente de OneDrive para la Empresa.

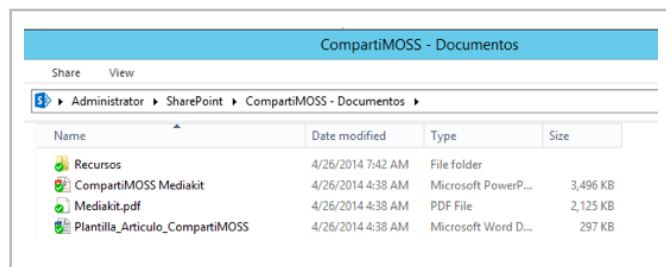


Imagen 3.- Archivos sincronizados localmente utilizando el cliente de OneDrive para la Empresa.

- Acciones de documento o carpeta, que incluyen las acciones “modificar”, “administrar” y “compartir”. La acción “modificar” permite editar el documento seleccionado con el programa cliente que corresponde (por ejemplo, para un documento Word se abrirá Microsoft Office Word para proceder a su edición). La opción “administrar” muestra para el documento seleccionado todas las opciones clásicas de menú de elemento de lista / biblioteca.

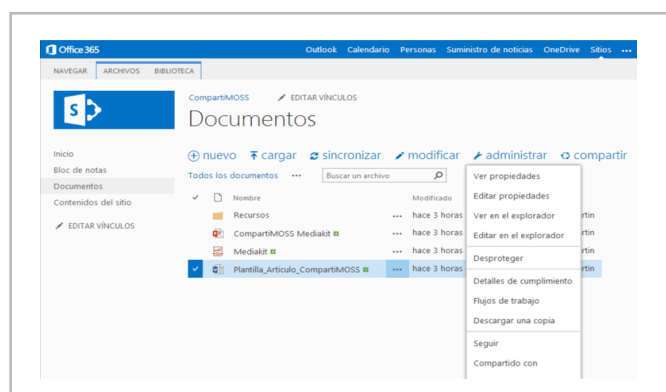


Imagen 4.- Opciones del menú de elemento de biblioteca desplegadas con la opción “administrar”.

Finalmente, la opción “compartir” abre el diálogo que permite compartir el documento seleccionado con otros usuarios del sitio o bien con usuarios externos de forma que se genere la correspondiente invitación de correo electrónico (si se deja esa opción marcada al compartir el documento).

Novedades en el acceso a documentos de SharePoint Online Junto con las novedades relativas a la gestión y uso de bibliotecas de documentos, Microsoft también ha incluido otras funciones que mejoran la experiencia en el acceso a documentos almacenados en bibliotecas de sitios de SharePoint Online:

- Cuando un usuario previsualiza un documento almacenado en una biblioteca de documentos, además de poder visualizarlo sin tener que abrirlo y de poder realizar acciones comunes de administración y uso, dispone de la posibilidad de facilitar el acceso al mismo desde un dispositivo móvil que disponga de una aplicación de escaneo de códigos QR.



Imagen 5.- Acceso al código QR del documento almacenado en SharePoint Online.

Como se puede apreciar en la Imagen 5, junto con la Url del documento en SharePoint Online se dispone de un icono que representa un dispositivo móvil.

- Si se hace clic en icono, a continuación se muestra una página con el código QR del documento así como la Url de acceso al mismo desde el navegador.

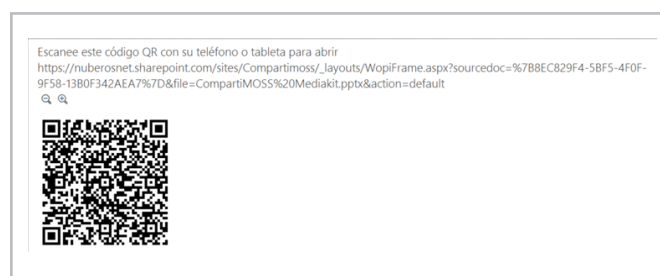


Imagen 6.- Código QR del documento.

- El código QR se puede escanear desde una aplicación que lea ese tipo de códigos en un dispositivo móvil. Por ejemplo, en mi caso, una aplicación específica para mi iPad Mini. Una vez escaneado el código, podemos acceder al documento desde el navegador del dispositivo móvil

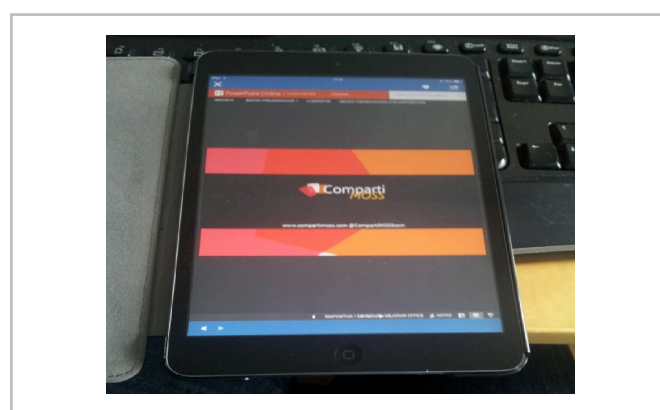


Imagen 6.- Acceso al documento almacenado en SharePoint Online haciendo uso del código QR en un dispositivo móvil.

JUAN CARLOS GONZÁLEZ MARTÍN

MVP de SharePoint Server

Arquitecto de Soluciones en LKS

E-Mail de Contacto: jc.gonzalez@lks.es

Twitter: @jcgm1978

Blog: <http://geeks.ms/blogs/jcgonzalez>

Web: www.lks.es

Buenas prácticas en infraestructura en SharePoint 2013 (Parte 2)

Resumen

En la segunda parte del artículo repasaremos las recomendaciones que hay que tomar cuando se diseña una granja SharePoint, los conceptos básicos de diseño, los parámetros en los que debemos basarnos para calcular el tamaño y capacidades y buenas prácticas a tener en cuenta a la hora de instalar y configurar nuestro entorno.

Artículo

Continuando el anterior artículo ([CompartiMOSS nº 19](#)), vamos a centrarnos en el diseño de una granja de SharePoint. Veremos las diferentes recomendaciones y buenas prácticas a la hora de diseñar la granja, calcular su tamaño y capacidades y configurar. Se ha intentado que el artículo aplique tanto a SharePoint 2010 como a SharePoint 2013, indicando en determinados puntos posibles diferencias. Como primera parte del artículo, es importante tener claro cómo funciona y se estructura un entorno SharePoint. Si ya conoces estos conceptos, puedes saltarte el primer apartado.

Escenario

Entendiendo cómo funciona un entorno SharePoint

Un conjunto de servidores (también llamado granja) puede contener uno o varios servidores de SharePoint. Los servidores de SharePoint se basan en IIS y en .NET para albergar las aplicaciones web de contenido y las aplicaciones de servicio. En numerosos diseños de arquitectura, los servidores SharePoint se separan en dos roles:

- Servidores frontal web: contienen las aplicaciones de contenido (una web, una intranet, los sitios personales, el repositorio documental, etc.).
- Servidores de aplicación: contienen las aplicaciones de servicio (Servicios de Access, servicio de administración de Apps, servicio de BCS, Excel Services, metadatos administrados, traducciones, PerformancePoint, búsquedas, servicio de Visio, servicio de perfiles, etc.).

Adicionalmente, es necesario tener uno o varios servidores SQL Server de base de datos, que albergarán todas las bases de datos de configuración, contenido y servicios de la granja. Opcionalmente se puede disponer de uno o varios servidores de Office Web Apps (también llamadas Office Online en entornos cloud), que proporcionan a SharePoint versiones

web de Word, Excel, PowerPoint y OneNote. En SharePoint 2010, las Office Web Apps se instalaban sobre SharePoint como aplicaciones de servicio adicionales, en la versión 2013, forman una granja totalmente independiente, que puede ser consumida por SharePoint o por otros productos de servidor.

En la Imagen 1 puede observarse la estructura de una granja SharePoint y cómo otros productos cliente y servidor interactúan con ella. Adicionalmente, Exchange Server y

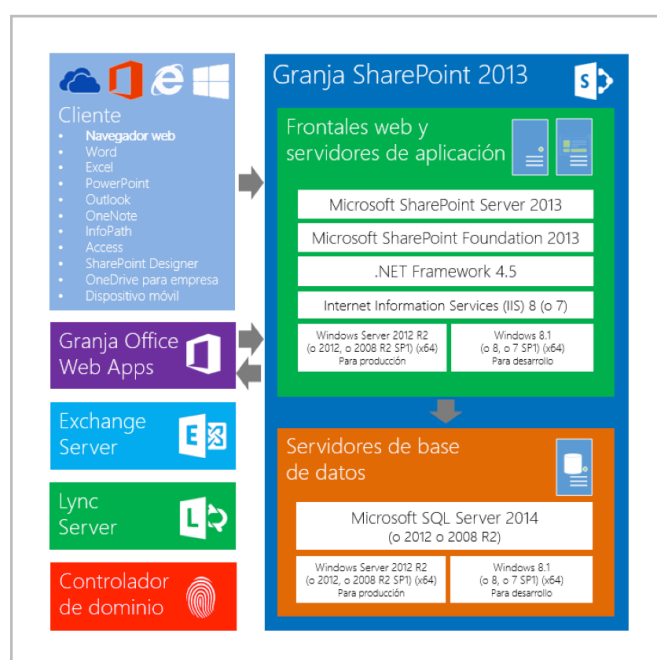


Imagen 1.- Estructura de un entorno SharePoint, incluyendo servicios y servidores adicionales.

Lync Server se pueden integrar en multitud de ocasiones con SharePoint, y son un complemento perfecto para su funcionalidad. Los ejemplos de integración más comunes son:

- Indicadores de presencia: cuando se visualiza un usuario en SharePoint, parece un indicador visual que permite conocer la presencia (desconectado, activo, ocupado...) de Lync. Además, el indicador permite enviar un correo electrónico con Exchange, iniciar un chat por Lync o realizar una videollamada.
- Buzón del sitio: la posibilidad de contar con un buzón de Exchange dentro de un sitio colaborativo de SharePoint para enviar y recibir correos. Los correos recibidos y sus adjuntos

pueden guardarse en las bibliotecas documentales de SharePoint.

- Work Management Service: aplicación de servicio que permite visualizar tareas de Exchange en SharePoint.
- Outlook Web App y Lync Web App: Exchange Server y Lync Server proporcionan versiones web de Outlook y Lync. Pueden mostrarse en SharePoint como WebParts.

¿Cómo estructurar una granja?

Cuando se diseña una granja de SharePoint, no existe una implementación única ni unas reglas estándar a seguir. Cada granja debe disponer de los componentes y servidores que más se adapten a nuestras necesidades. SharePoint es un producto que permite mucha flexibilidad a la hora de separar funcionalidades y servicios según nos interese.

Así mismo, no hay un modelo estándar de cálculo de capacidades. Debemos hacer pruebas de capacidad con cálculos sobre papel y fijarnos en otras granjas y arquitecturas de referencia para calcular los recursos necesarios. Una vez diseñada e implementada la granja, será necesario realizar pruebas de carga y evaluar si la granja satisface nuestras necesidades o si es necesario un cambio o re-escalado.

Para ello, Microsoft propone un modelo cíclico (Imagen 2) en el que modelamos y diseñamos la granja en base a nuestros datos de rendimiento, carga y funcionalidades. Tras esto, implementamos un piloto en el que probamos los recursos y la carga del hardware. Si es necesario, se optimizará la granja realizando los cambios necesarios en los recursos de la granja y la topología. Una vez se haya terminado, se implementará la granja final y se pasará a un estado de supervisión y mantenimiento, en la que se monitorizará la granja para ir evaluando periódicamente si es necesario aumentar sus capacidades. En caso que sea necesario, se comenzará de nuevo con el proceso. Se puede encontrar más detalle de este proceso en: [http://technet.microsoft.com/es-es/library/ff758647\(v=office.15\).aspx](http://technet.microsoft.com/es-es/library/ff758647(v=office.15).aspx)

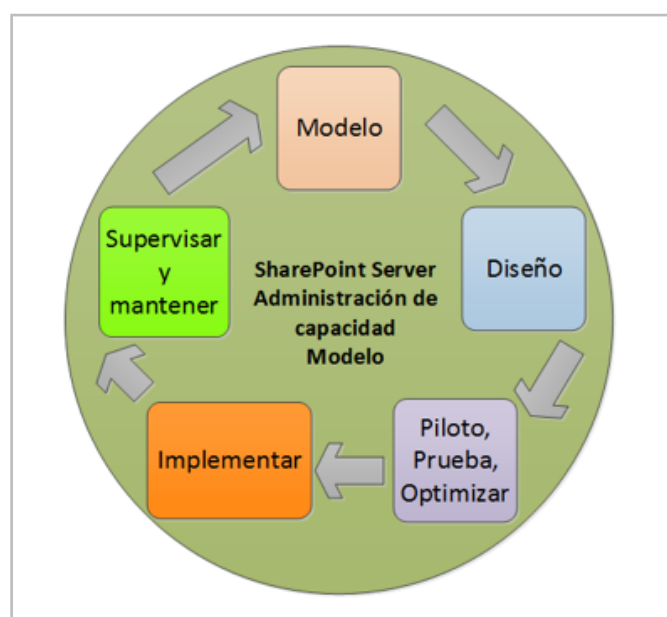


Imagen 2.- Modelo cíclico.

Diseño de la granja

Para diseñar la granja, es necesario conocer las distintas combinaciones de servidores con las que podemos ir jugando para adaptarla a nuestras necesidades. Comencemos con lo básico, el mínimo de servidores en producción son: un servidor SharePoint (que contendrá aplicaciones web de contenido y aplicaciones de servicio) y un servidor de base de datos (Imagen 3).

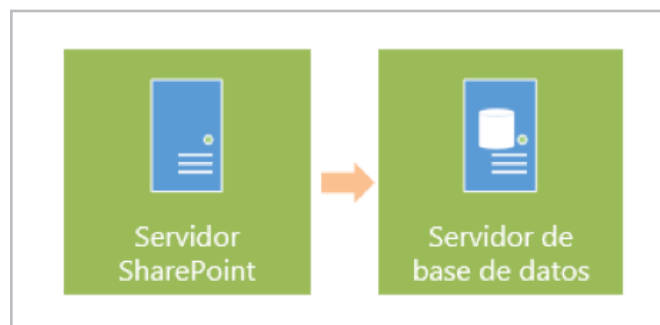


Imagen 3.- Mínimo de servidores para un entorno de producción.

Si el servidor tiene mucha carga de datos, o mucho acceso y concurrencia, es posible agregar más servidores (los que sean necesarios). El proceso es tan sencillo como instalar SharePoint en un nuevo servidor y ejecutar un asistente para unirlos a la granja. En estos casos, podemos balancearlos (usando NLB de Windows u otra tecnología) para repartir la carga entre ellos, de esta forma también conseguimos contar con Alta Disponibilidad (Imagen 4).

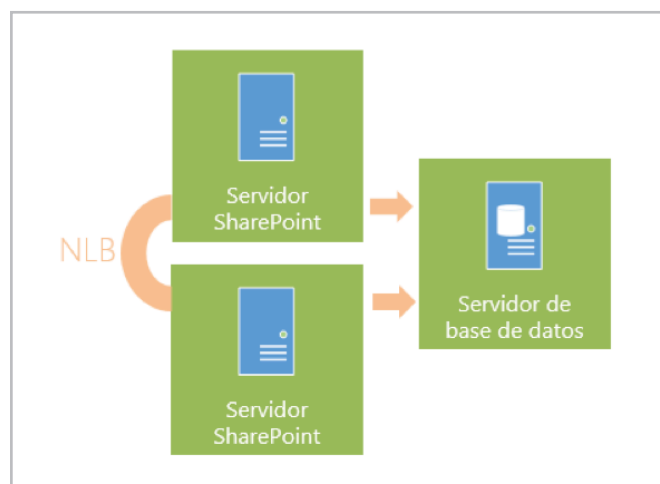


Imagen 4.- Podemos añadir los servidores de SharePoint que sean necesarios y balancearlos.

Para conseguir Alta Disponibilidad en los servidores de base de datos, podemos añadir más servidores y configurarlos en SQL Server Cluster o en SQL Server Always-on (Imagen 5).

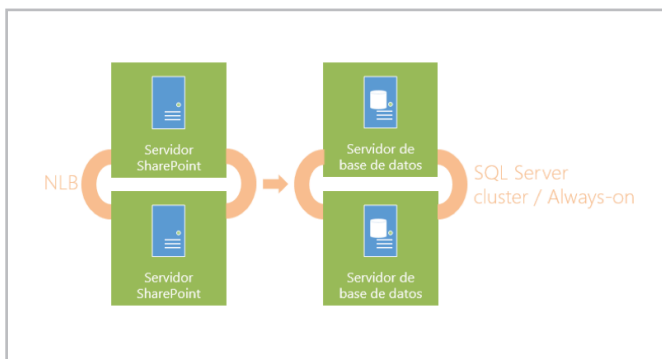


Imagen 5.- También podemos añadir más servidores SQL Server para Alta Disponibilidad.



Imagen 6.- Podemos diferenciar entre servidor frontal web y servidor de aplicación.

Es posible añadir la cantidad de servidores SharePoint que sea necesario (dos, tres, diez, treinta...) para cubrir nuestras necesidades. Del mismo modo, podemos separar en servidores de aplicación las aplicaciones de servicio que consideremos oportunas y hacer los balanceos que sean necesarios. Cuando tenemos entornos con varios servidores frontal web y varios de aplicación, tenemos una de las arquitecturas más famosas, la arquitectura de tres niveles. En la Imagen 7 podemos ver un ejemplo de arquitectura de tres niveles con Alta Disponibilidad.

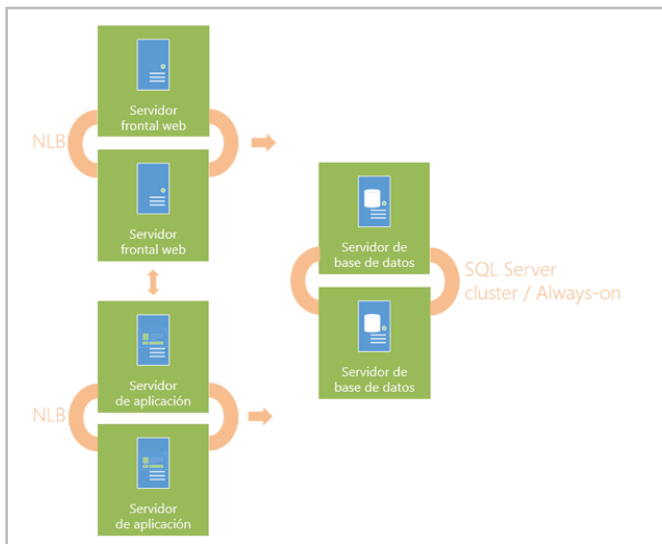


Imagen 7.- Arquitectura de tres niveles (web, aplicación y base de datos) con Alta Disponibilidad.

Cálculo del tamaño

El siguiente paso, una vez conocidos estos conceptos, es calcular el tamaño que tendrá la granja. Para ello, Microsoft nos propone cuatro arquitecturas de referencia, para que nos basemos en la que más se adapta a nuestras necesidades. Estas cuatro arquitecturas son:

- Granja de servidor único (Imagen 8):
 - Para fines de evaluación.
 - Todo (frontales web, servidores de aplicaciones, bases de datos) se instala en un único servidor.
- Granja de servidores pequeña (Imagen 8)
 - También llamada “granja de dos niveles”.
 - Para implementaciones limitadas, con un mínimo de aplicaciones de servicio en uso (o muchas con poca carga) y un volumen relativamente pequeño de datos (10 o más GB).
- Granja de servidores mediana (Imagen 9)
 - También llamada “granja de tres niveles”.
 - Tiene una topología de tres niveles, (la arquitectura más común) en la que diferenciamos servidores frontal web (aplicaciones de contenido), servidores de aplicación (aplicaciones de servicio) y servidores de base de datos.
 - Decenas de miles de usuarios.
 - De 10 a 50 solicitudes por segundo.
 - 1 ó 2 TB de datos.
- Granja de servidores grande (Imagen 10):
 - Consiste en varias granjas de servidores, en las que podemos o no compartir aplicaciones de servicio entre ellas.
 - De esta forma, podemos conseguir granjas dedicadas para distintos servicios. Podemos tener una granja sólo para servicios de Excel que sea consumida por otras granjas.
 - Cientos de miles de usuarios.
 - Cientos de solicitudes por segundo.
 - 10 o más TB de datos.



Imagen 8.- Granja de servidor único y granja de servidores pequeña.



Imagen 9.- Granja de servidores mediana.

Cuando se tiene clara la arquitectura de referencia con la que se va a trabajar y se tienen claros los conceptos de diseño, es el momento de calcular las capacidades de la granja. Para ello, debemos recopilar toda la información que podamos, como la posible carga del entorno, cantidad de usuarios, volumen de los datos, valores de concurrencia

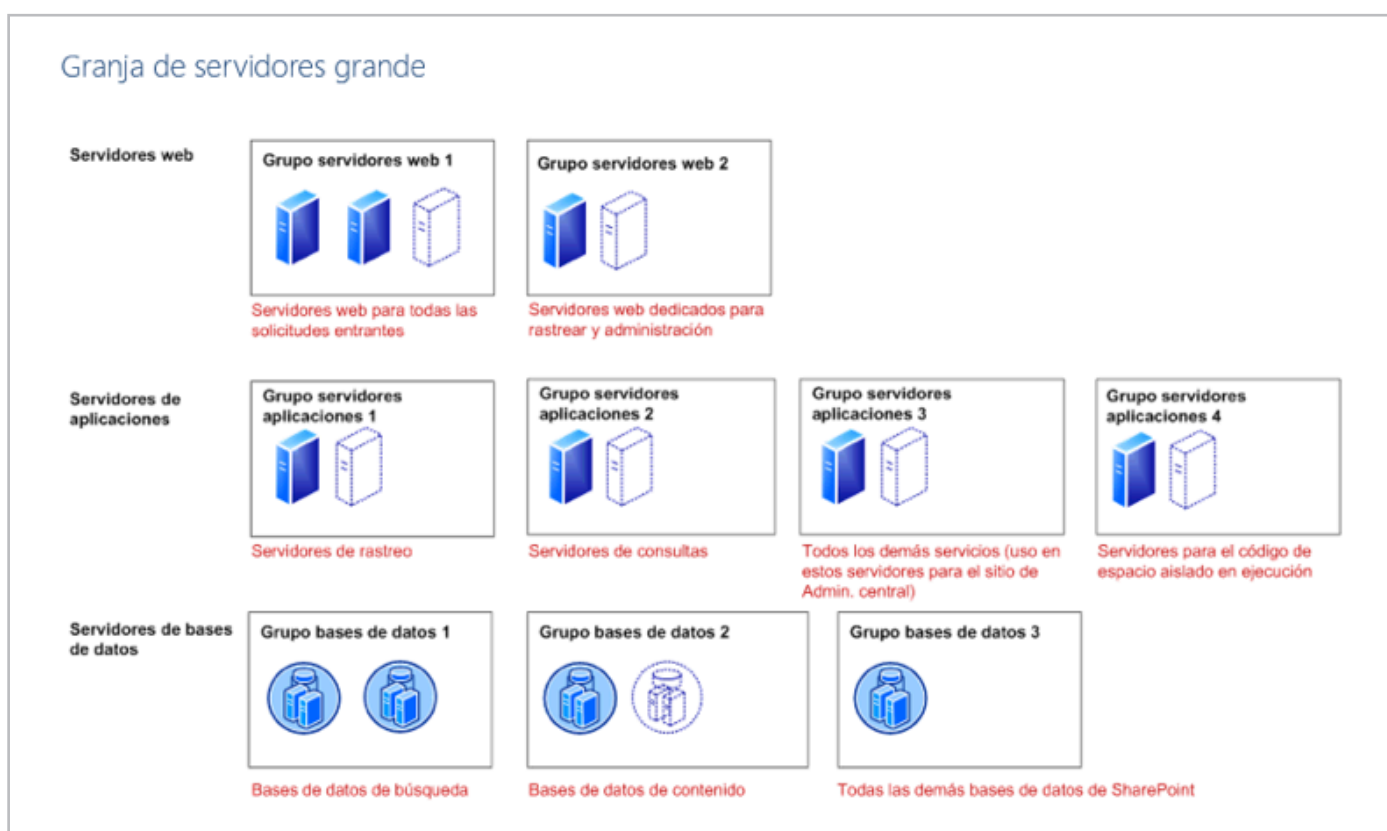


Imagen 10.- Granja de servidores grande

Estas arquitecturas sirven de referencia, no son un modelo cerrado que debamos implementar. Debemos basarnos en ellas y hacer los cambios que consideremos oportunos para diseñar nuestro propio entorno. Se puede encontrar más detalle sobre estas arquitecturas de referencia en el siguiente enlace: [http://technet.microsoft.com/es-es/library/ff758647\(v=office.15\).aspx](http://technet.microsoft.com/es-es/library/ff758647(v=office.15).aspx)

Cálculo de la capacidad

Cuando se tiene clara la arquitectura de referencia con la que se va a trabajar y se tienen claros los conceptos de diseño, es el momento de calcular las capacidades de la granja. Para ello, debemos recopilar toda la información que podamos, como la posible carga del entorno, cantidad de usuarios, volumen de los datos, valores de concurrencia, etc.

Para realizar este cálculo, debemos utilizar un importante recurso de TechNet, los resultados y recomendaciones de pruebas de rendimiento (<http://technet.microsoft.com/es-es/library/ff608068.aspx>). Este artículo contiene pruebas de rendimiento, recomendaciones y arquitecturas de ejemplo para diferentes escenarios (colaboración, ECM, contenido multimedia, contenido social...). Podemos consultar el escenario que más se adapte a nuestro caso y hacer los cálculos de rendimiento para nuestra granja basándonos en las indicaciones.

Adicionalmente, es recomendable diseñar las capacidades basándonos en la infraestructura diseñada en proyectos pasados y en los ejemplos reales de implementaciones que se pueden encontrar en TechNet.

También existen calculadora automáticas que pueden ayudarnos en muchos casos:

- HP SharePoint Sizer (para SharePoint 2007 y 2010):
- <http://h71019.www7.hp.com/ActiveAnswers/secure/548230-0-0-225-121.html>
- Microsoft SharePoint Capacity Planning Tool (para SharePoint 2007):
- <http://www.microsoft.com/en-us/download/details.aspx?id=22876>

Finalmente, este artículo de Sanjay Narang puede ayudarnos mucho para calcular capacidades en SharePoint 2013: <http://blogs.msdn.com/b/sanjaynarang/archive/2013/04/06/sizing-and-capacity-planning-for-sharepoint-2013-resources.aspx>

Otras recomendaciones

Finalmente, no quería terminar el artículo sin detallar otras recomendaciones a tener en cuenta al configurar o implementar granjas de SharePoint. La primera de ellas son las cuentas de usuario. Nunca queda claro del todo cuales son las cuentas necesarias y recomendadas para configurar al completo la granja, ya que Microsoft las va detallando a lo largo de diferentes artículos de TechNet y puede llevar a cierta confusión. A continuación os indico el listado completo, que es compatible con proyectos de instalación como AutoSPInstaller. Existen distintas nomenclaturas, en este listado he intentado usar una de las más extendidas, que consiste en poner las letras “sp” delante de cada cuenta, seguido de una explicación breve de su función:

- spsetup: Cuenta de instalación. Con ella instalamos SharePoint y ejecutamos el asistente de configuración de la granja. Será la cuenta apropiada para la ejecución de PowerShell.
- Debe ser administrador local en todos los servidores, incluyendo SQL Server. Debe tener los permisos securityadmin y dbcreator en SQL Server.
- spfarm: cuenta de acceso a base de datos, administrador de la granja, pool de la administración central y servicio de timer. Se impersona en la granja como “Cuenta del Sistema” (SHAREPOINT\system). Es la cuenta que debemos escribir para conectar a base de datos en el asistente de configuración de la granja.
- Debe ser administrador local en todos los servidores SharePoint durante la instalación y para arrancar el servicio de sincronización de perfiles. Una vez acabemos estas tareas NO debe ser administrador local nunca más.
- sppool1, sppool2, sppool3...: cuentas para los application pool de las distintas aplicaciones SharePoint de contenido. Una cuenta por aplicación.
- spmysite: administrador de Mi Sitio (My Site) y cuenta del application pool de la aplicación de Mi Sitio.
- Debe ser administrador local en todos los servidores SharePoint.

- spservices: cuenta para los application pool de las aplicaciones de servicio
- spproject: si se instala Project Server, se recomienda una cuenta adicional para el application pool de las aplicaciones de servicio de Project Server.
- spsearch: cuenta para el servicio de búsqueda.
- spcrawl: cuenta para el rastreo de contenido en las búsquedas.
- spprofile: cuenta para sincronizar perfiles de usuario.
- Debe tener permisos de “Replicating Directory Changes” en el Active Directory. Si el dominio es versión Windows 2003 o anterior, debe ser miembro del grupo “Pre-Windows 2000” del Active Directory.
- spexcel, spvisio, spperfpoin: cuentas para el servicio sin-
atención de los servicios de Excel, Visio y PerformancePoint.

- spcacheadm, spcacherd: cuentas para la caché de objetos de SharePoint (<http://technet.microsoft.com/es-es/library/ff758656.aspx>)

Nota: Hay cuentas en las que no se indica los permisos necesarios. Esto es debido a que se le asignarán automáticamente durante la instalación o configuración.

A continuación, una serie de recomendaciones a tener en cuenta en los servidores de base de datos:

- A ser posible, conectar a base de datos mediante un Alias SQL. Si existe la necesidad de cambiar el servidor de base de datos, de configurar un cluster o de realizar cualquier otro cambio, SharePoint no se verá afectado.
- Si se dispone de servidores suficientes, configurar clúster en base de datos (o un Always-on) para conseguir alta disponibilidad.
- Separar bases de datos en distintos discos duros. Recomendación:
- Disco 1: Sistema Operativo
- Disco 2: Binarios, instalación
- Disco 3: bases de datos de contenido y servicios (mdf)
- Disco 4: log transaccionales de bases de datos de contenido y servicios (ldf)
- Disco 5: bases de datos de búsqueda y rastreo (mdf)
- Disco 6: log transaccionales de bases de datos de búsqueda y rastreo (ldf)
- El Autogrow de bases de datos está por defecto a 1 MB, mejor incrementarlo.
- Intentar no cambiar permisos en las bases de datos de SharePoint a mano, a no ser que sea absolutamente necesario. Para dar permisos a cuentas adicionales, usar el comando de PowerShell Add-SPShellAdmin desde los servidores SharePoint.

Y otra serie de recomendaciones en los servidores SharePoint:

- Crear aplicaciones de servicio a mano para controlar los nombres de las bases de datos, pools de aplicaciones, configuraciones y cuentas.
- A ser posible, conectar a base de datos mediante un Alias SQL.
- Configurar límites de ficheros de log, separar en otros discos duros.
- Instalar iFilters esenciales, p.ej. PDF.
- Al acabar la instalación, spfarm debe dejar de ser administrador local.
- Si se dispone de servidores suficientes, configurar balanceos NLB en los frontales para lograr Alta Disponibilidad.
- Si se desea, configurar una granja de Office Web Apps y conectarla a la granja de SharePoint.

- Configurar correctamente los servicios que se ejecutan en cada servidor, es un error común hacer servidores de aplicaciones con el servicio de frontal web. Evitar esto.
- Supervisar periódicamente la granja e instalar actualizaciones de seguridad si es necesario.
- Instalar los Service Pack cuando vayan apareciendo, aunque recomiendo esperar unas semanas por si se descubre algún problema.

Conclusiones

Espero que estos dos artículos puedan servir de ayuda a la hora de diseñar e implementar entornos de SharePoint. Existe muchísima información sobre estos procesos en internet, pero toda muy desordenada y extremadamente densa. No olvidéis todas las recomendaciones, ya que evitarán muchos problemas y quebraderos de cabeza en el futuro.

MIGUEL TABERA PACHECO

SharePoint Server MVP

miguel.tabera@outlook.com

[@migueltabera](https://www.sinsharepointnohayparaíso.com)

<http://www.sinsharepointnohayparaíso.com>

KWizCom Forms

Formularios Nativos para SharePoint y Soluciones Móviles

Cree fácilmente formularios personalizados utilizando los formularios de las listas existentes. Sin necesidad de despliegue ni aprender a utilizar nuevas herramientas!

www.kwizcom.com/forms



Exponiendo datos binarios con BCS (II)

Resumen

En este artículo veremos cómo exponer a SharePoint datos binarios de otros sistemas a través de Business Connectivity Services (BCS), utilizando el conector de tipo ensamblado de NET.

Artículo

Business Connectivity Services (BCS) es un componente de SharePoint que permite exponer datos provenientes de otros sistemas (bases de datos, aplicaciones corporativas, servicios online...) dentro de SharePoint como listas externas.

En el artículo anterior de esta serie, publicado en CompartiMOSS número 18, expliqué como acceder a los datos binarios que residen en una tabla de SQL Server. Para ello, bastaba con apuntar SharePoint Designer a dicha tabla y retocar ligeramente el fichero descriptor del conector que genera la herramienta.

Ahora vamos a ver cómo podemos exponer los datos en cualquier plataforma accesible mediante .NET como lista externa en SharePoint. Para ello tendremos que usar nuestro querido amigo Visual Studio.

Conector BCS usando Visual Studio

Cuando queremos hacer un conector de BCS y los datos a los que apuntamos no están en SQL Server u otra base de datos compatible con ODBC, no hay más remedio que conectarse mediante un conector de tipo ensamblado (Assembly). El conector realmente será una librería NET que expondrá algunos métodos para devolver los datos del sistema externo.

Como punto de partida en este ejemplo tendremos un ensamblado que devolverá una lista de una imagen en formato JPG. Para simplificar el desarrollo, lo que haremos es fijar los contenidos de esta lista directamente en el código, a modo hardcoded. Dejaré

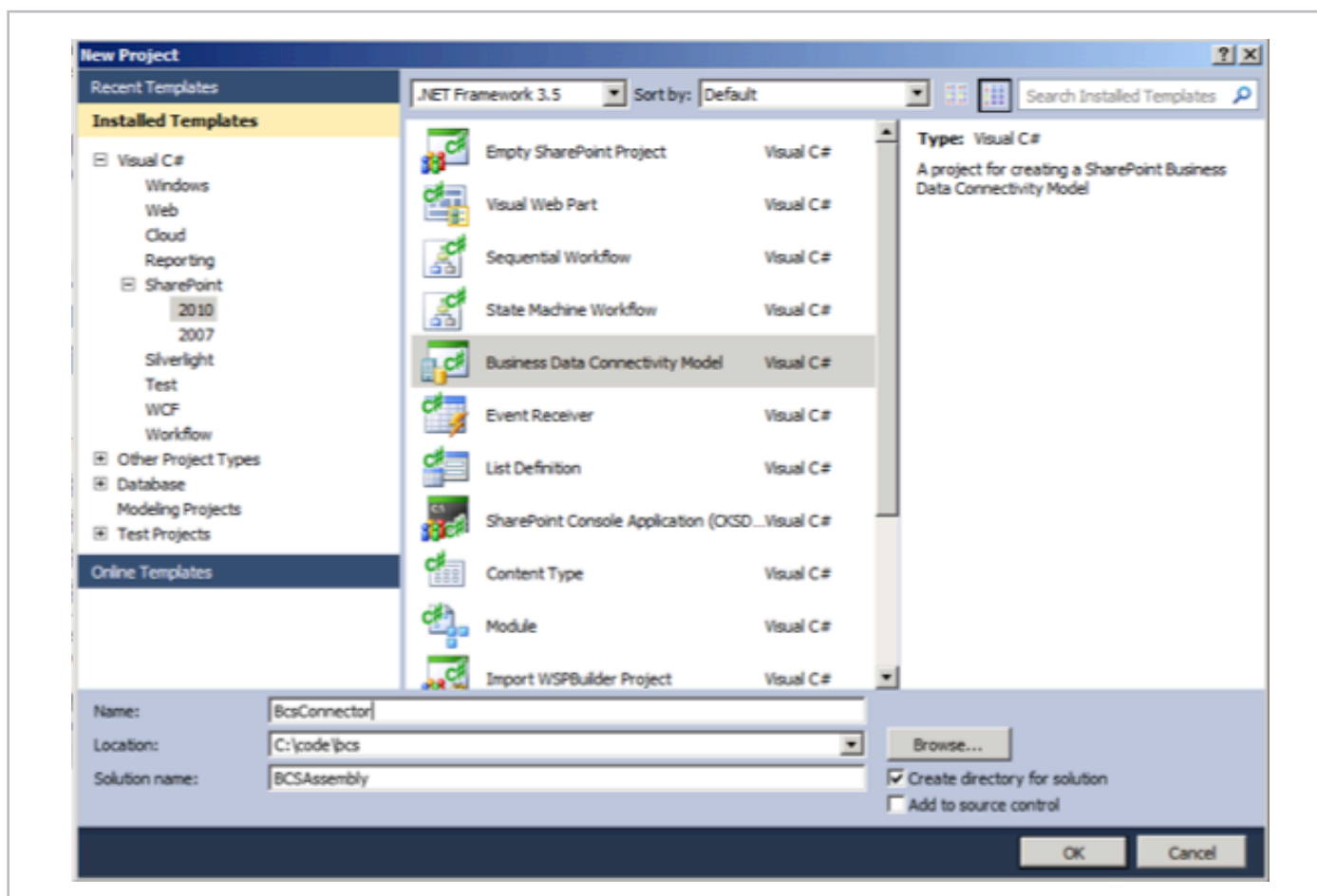


Imagen 1.- Nuestro proyecto de conector BCS como ensamblado NET.

como ejercicio al lector extender este ejemplo para que los datos vengan de una base de datos, un servicio web o cualquier otra fuente.

Abrimos Visual Studio (en mi ejemplo usaré Visual Studio 2010, pero la verdad es que el diseñador no difiere en nada en Visual Studio 2012 y 2013) y creamos un nuevo proyecto de tipo Modelo de conectividad a datos empresariales.

El proyecto lo he llamado BcsConnector dentro de la solución BCSAssembly.

Después de la obligada ventana de elegir dónde vamos a desplegar nuestro paquete de solución WSP, nos aparece la pantalla siguiente.

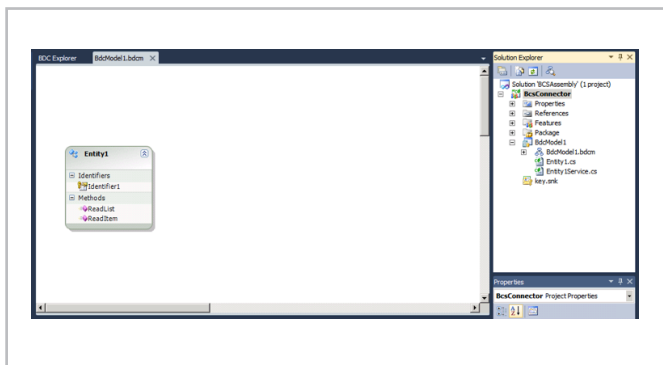


Imagen 2.- El conector BCS autogenerado por Visual Studio.

Podemos ver que Visual Studio ha creado un conector automático, inventándose una entidad llamada Entity1 y un modelo llamado BdcModel1. Vale la pena detenernos aquí y explicar el significado de cada uno de los ficheros generados.

BdcModel1.bdcml es el fichero del descriptor del modelo BCS. Es un fichero XML y contiene la descripción de nuestro modelo, los metadatos que contienen las entidades del modelo y las operaciones que devuelven los metadatos. Como se puede ver, Visual Studio dispone de un diseñador visual del modelo, al estilo de Entity Framework.

Entity1.cs es la clase que representa nuestro modelo de datos. Si cambiamos su nombre o sus propiedades, el fichero descriptor no se actualiza automáticamente. Esto es muy importante porque muchos fallos de BCS vienen precisamente de la falta de sincronización entre el ensamblado y el descriptor.

Entity1Service.cs es la clase que implementa el servicio que devuelve los datos. En esta clase se generarán los métodos que devuelven los datos. Si en el diseñador añadimos o cambiamos un método, ese cambio se verá reflejado en esta clase así que habrá que ir alerta con los cambios no deseados en el diseñador.

Bien, ahora vamos a renombrar la Entity1 como Photo y Entity1Service como PhotoService. A la clase Photo le vamos a poner las siguientes propiedades.

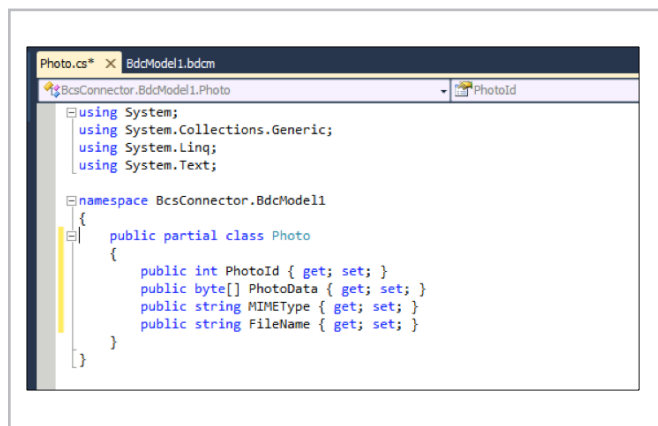


Imagen 3.- La clase Photo.

PhotoId será la columna clave o identificador de cada foto (aunque en el ejemplo tenemos sólo una foto). PhotoData contendrá los bytes con los datos binarios de la foto. MimeType tendrá el texto de tipo de contenido MIME (en nuestro caso "image/jpeg") y FileName contendrá el nombre del fichero que se descargará desde nuestro conector. PhotoData, MimeType y FileName son necesarios para poder descargar los datos binarios y un identificador es necesario para que BCS sepa cómo acceder a los detalles de una entidad.

Si abrimos el diseñador del modelo y vamos a la pestaña BDC Explorer, podemos ver que nuestro modelo real y el modelo en el descriptor no coinciden.

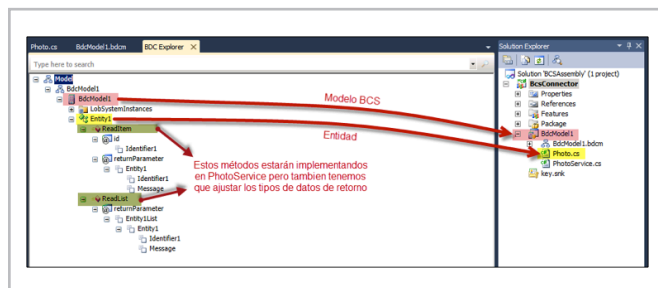


Imagen 4.- El descuadre entre el modelo y el ensamblado.

Vemos que el modelo todavía refleja la Entity1 y que las operaciones ReadItem y ReadList (las dos mínimas que necesita BCS) devuelven listas e instancias de Entity1. Tenemos que arreglar esto, clicando en cada nodo y especificando el tipo de datos correcto.

Primero renombraremos la entidad Entity1 en el modelo (el nodo destacado en amarillo) a Photo. Para ello, clicaremos el nodo y en la ventana de Propiedades ajustaremos en nombre.

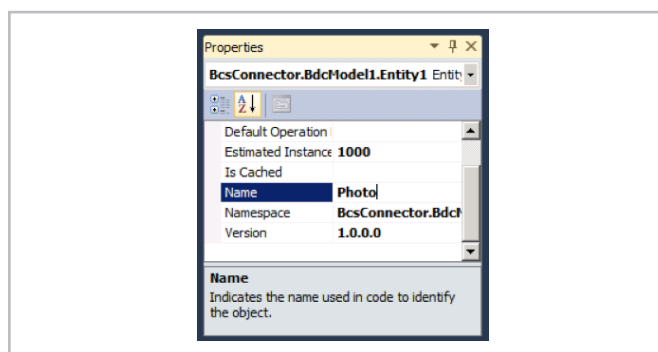


Imagen 5.- Ajustando los metadatos del modelo.

Acto seguido, abriremos el diseñador BCS y en la entidad Photo le cambiaremos el nombre y el tipo de identificador, de Identifier1 (String) a PhotoId (Int32).

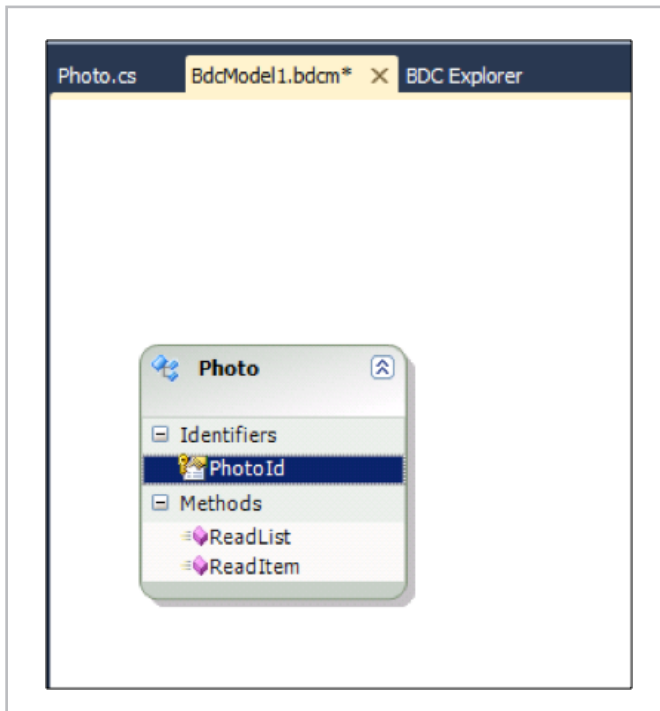


Imagen 6.- Corrigiendo el identificador de la entidad Photo.

Ahora viene la parte más manual de todo el proceso. En el BDC Explorer tenemos que ir a cada método (ReadList y ReadItem) y cambiar manualmente los tipos de datos de los parámetros de entrada y salida.

ReadItem tomará como parámetro id de tipo Int32, y devolverá un nodo de tipo Photo, con los subnodos PhotoId (Int32), FileName (String) y MIMEType (String). Clicando con el botón derecho encima de un nodo de resultado y eligiendo "Add Type Descriptor" podemos ir pegando las propiedades de nuestra entidad.

Para elegir el tipo de entidad Photo, en la ventana propiedades tenemos una pestaña para elegir las entidades del proyecto actual.

ReadList no tendrá parámetros pero devolverá un IEnumerable de Photo llamado PhotoList (hay que marcar la casilla "Is Enumerable" al elegir el tipo), con los mismos parámetros que ReadItem. Para ello, podemos copiar el nodo Photo entero desde ReadItem y pegarlo en ReadList, debajo del nodo de IEnumerable.

Todo el modelo queda ahora como se ve en la imagen siguiente.

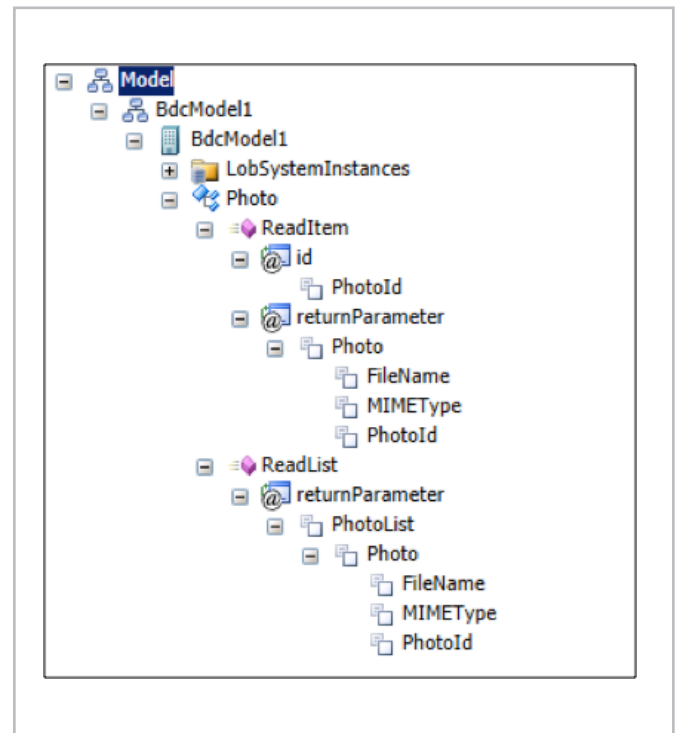


Imagen 8.- El modelo completo.

La parte más quisquillosa de hacer un conector está hecha. Nos faltan todavía dos cosas más: añadir el método que devolverá la foto (un StreamAccessor) y añadir el código que implementará los métodos ReadList y ReadItem.

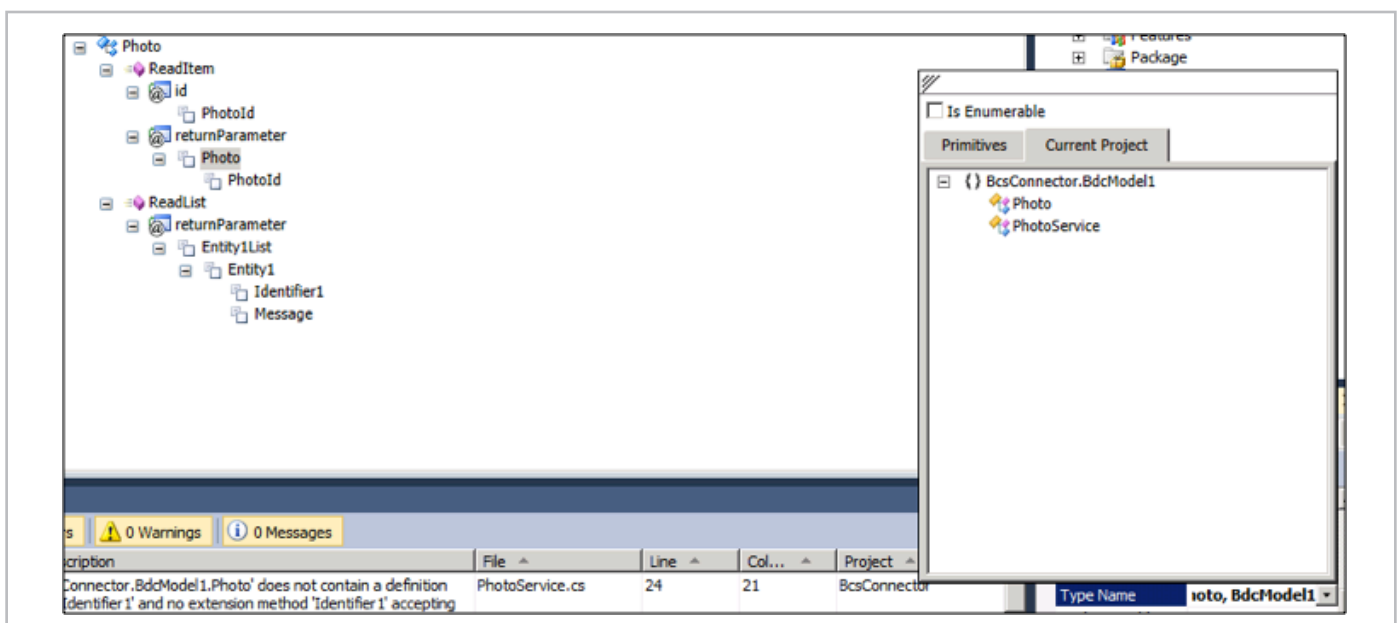


Imagen 7.- Especificando los tipos de datos para los métodos de BCS.

Para añadir el método StreamAccessor, tenemos que abrir el fichero de descriptor (BdcModel1.bdcmodel) en modo XML. Buscaremos el nodo correspondiente al método ReadItem y al final de este copiaremos el fragmento siguiente (antes del cierre del nodo Methods):

```
<Method Name="ReadPhoto" IsStatic="false">
  <Parameters>
    <Parameter Name="PhotoId" Direction="In">
      <TypeDescriptor Name="PhotoId" TypeName="System.Int32"
IdentifierEntityNamespace="BcsConnector.BdcModel1" IdentifierEntityName="Photo"
IdentifierName="PhotoId" />
    </Parameter>
    <Parameter Name="photo" Direction="Return">
      <TypeDescriptor Name="photoTypeDescriptor" TypeName="System.Stream" />
    </Parameter>
  </Parameters>
  <MethodInstances>
    <MethodInstance Name="ReadPhotoInstance" Type="StreamAccessor"
ReturnParameterName="photo" Default="true" DefaultDisplayName="View Photo"
ReturnPropertyDescriptorPath="photoTypeDescriptor">
      <Properties>
        <Property Name="MimeTypeField" Type="System.String">MimeType</Property>
        <Property Name="FileNameField" Type="System.String">FileName</Property>
      </Properties>
    </MethodInstance>
  </MethodInstances>
</Method>
```

Este fragmento define una operación llamada ReadPhoto que devolverá un Stream con la imagen, dado un PhotoId. Ahora ya estamos listos para preparar el código de PhotoService, que contendrá la lógica de los tres métodos (ReadItem, ReadList y ReadPhoto). Como podemos ver, estamos “falseando” la lista a mano pero para los propósitos de este artículo ya es suficiente.

01 CORPORATIVO <small>Identidad corporativa, logotipo, manual de identidad de marca, folletería, tarjetas de presentación, etc.</small>	+	DISEÑO CORPORATIVO Logotipo Manual de identidad de marca Folletería
02 DISEÑO WEB <small>Diseño de interfaces web, maquetación HTML + CSS, diseño de ecommerce, diseño de CRM, etc.</small>	▶	DISEÑO WEB Maquetación HTML + CSS Diseño de ecommerce Diseño de CRM
03 IMPRESO <small>Diseño editorial, diseño de packaging, diseño de urbanización, etc.</small>		DISEÑO IMPRESO Diseño Editorial Diseño Packaging Diseño de Urbanización
04 PUBLICITARIO <small>Diseño de avisos publicitarios, newsletters, banners, etc.</small>	●	DISEÑO PUBLICITARIO Avisos publicitarios Newsletter Banners


GRAPHIC & WEB DESIGN
 Peatonal Sarandí 975, piso 2 of 18
 Tel: +598 2916 7559
www.nitido.com.uy
contacto@nitido.com.uy

```

public static Photo ReadItem(int id)
{
    return new Photo()
    {
        PhotoId = id,
        FileName = "compartimoss.jpg",
        MIMType = "image/jpeg"
    };
}

public static IEnumerable<Photo> ReadList()
{
    List<Photo> list = new List<Photo>();
    var photo = new Photo()
    {
        PhotoId = 1,
        FileName = "compartimoss.jpg",
        MIMType = "image/jpeg"
    };
    list.Add(photo);
    return list;
}

public Stream ReadPhoto(int PhotoId)
{
    string imageString = "/9j/4AAQ5...kZJRgA";
    var photoData = System.Convert.FromBase64String(imageString);
    return new MemoryStream(photoData);
}

```

En imageString guardaremos los datos binarios de la imagen en formato Base64. Podéis usar la página <http://www.base64-image.de/step-1.php> para subir una imagen y obtener el texto en Base64 con los datos binarios de la imagen. En el código anterior he recortado la longitud del string pero debéis pegar el texto correspondiente a una imagen JPEG válida antes de probar el conector. No nos queda más que compilar, desplegar la solución, crear una lista externa de este tipo llamada Fotos y añadir una WebPart de tipo Lista de datos de conectividad empresarial. Nos debería salir la lista de una sola foto con el enlace para descargarla. Si lo clicamos veremos que la foto se descarga en toda su gloria binaria.

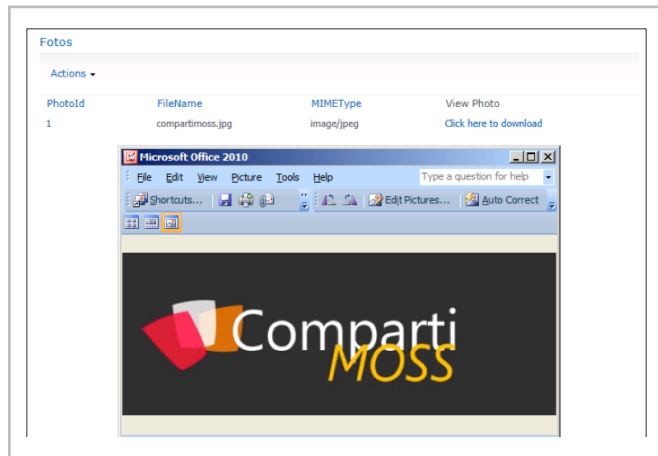


Imagen 9.- La satisfacción final de trabajo bien hecho.

Conclusión

En este artículo hemos visto cómo acceder a los datos de tipo binario desde SharePoint a través de un ensamblado .NET. Espero que los dos artículos hayan sido útiles para aclarar un poco que pasa cuando trabajamos con datos binarios en BCS. Como decía en el primer artículo, es un tema muy poco cubierto por la documentación. El fichero modelo BCS de este artículo está disponible en <http://1drv.ms/1g0INMP>.

EDIN KAPIC

Arquitecto SharePoint

edin.kapic@spenta.es

ekapic / <http://www.spenta.es>



Expertos en SharePoint

Soluciones
integrales



Queremos ser su socio tecnológico

Windows 8

SharePoint

Windows Phone

Windows Azure



comercial@gsc.es

www.gsc.es

Páginas Maestras y Aplicaciones SharePoint-Hosted

Resumen

En este artículo veremos cómo podemos especificar nuestra página maestra personalizada en una Aplicación SharePoint-Hosted, así como algunas consideraciones a tener en cuenta al respecto.

Artículo

Desde la llegada de las Aplicaciones de SharePoint y sus tres modelos de hosting posibles: SharePoint-Hosted, Autohosted y Provider Hosted, el modelo de las Aplicaciones SharePoint-Hosted es probablemente uno de los que mejor acogida ha tenido, sobre todo para el desarrollador, gracias a que es mucho más sencillo de desplegar en SharePoint.

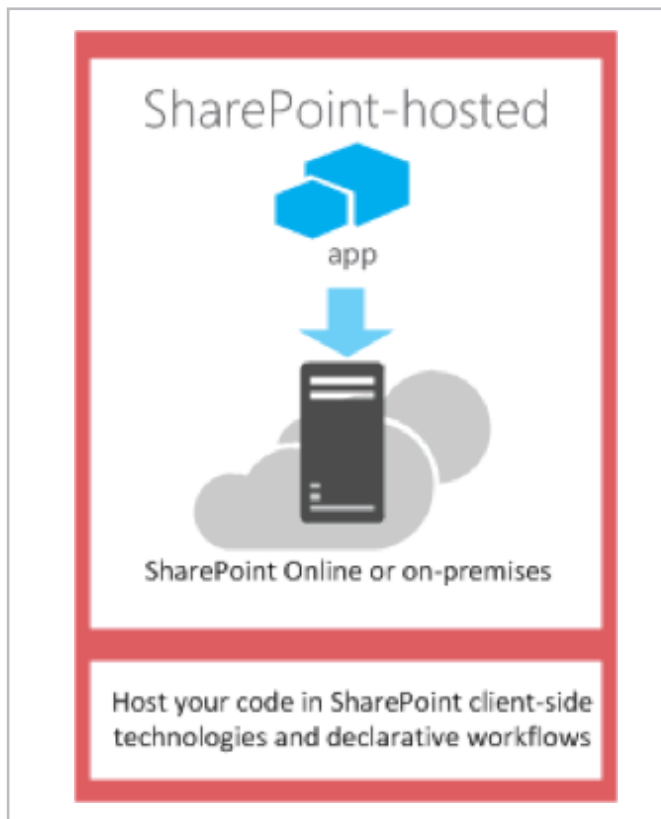


Imagen 1.- Arquitectura de una Aplicación SharePoint-Hosted.

Sin embargo, no olvidemos que cualquier App se ejecuta fuera de SharePoint, incluso las SharePoint-Hosted, y esto trae algunos desafíos relacionados con el branding, a la hora de mantener una estética coherente entre el sitio de SharePoint y la App.

Cuando instalamos una Aplicación SharePoint-Hosted, SharePoint provisiona la App en un sitio específico que es creado en el momento en el que se instala la App desde el sitio Host, con una URL característica que seguramente ya todos conocéis:

<http://app-bdf2016ea7dacb.fabrikamapps.com/sites/Marketing/Scheduler>

[https:// App_Prefix - App_ID . App_Base_Domain / Domain_Relative_URL_of_Host_Web / App_Name](https://App_Prefix - App_ID . App_Base_Domain / Domain_Relative_URL_of_Host_Web / App_Name)

Además, SharePoint provisiona la galería de Páginas Maestras, incluyendo algunas de las páginas maestras por defecto de SharePoint, tal y como se puede apreciar en la Imagen 2.

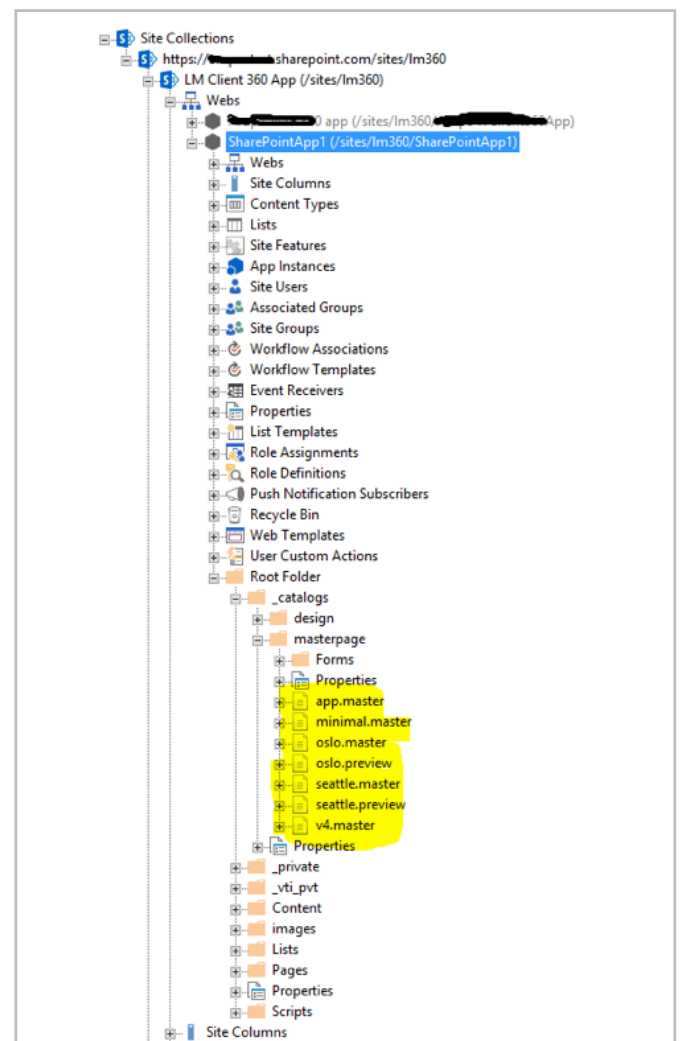


Imagen 2. Master page gallery en una SharePoint hosted app

Conviene aclarar que SharePoint provisiona algunas de las páginas maestras que tiene por defecto, y que en ningún caso provisiona las mismas páginas maestras que contiene el sitio de SharePoint donde se instaló la App.

Por defecto, SharePoint establece como página maestra la página “app.master”, dicha página está preparada para aplicar algunos de los estilos de SharePoint, así como el Chrome control, que hará posible que el tema seleccionado en el sitio de SharePoint donde se instaló la App. Si observamos la imagen 3, vemos uno de los temas que viene por defecto en SharePoint, aplicado a un sitio, en dicho sitio hemos instalado una Aplicación SharePoint-Hosted, que utiliza la página maestra app.master, y tiene el mismo tema aplicado, como se puede observar en la Imagen 4.

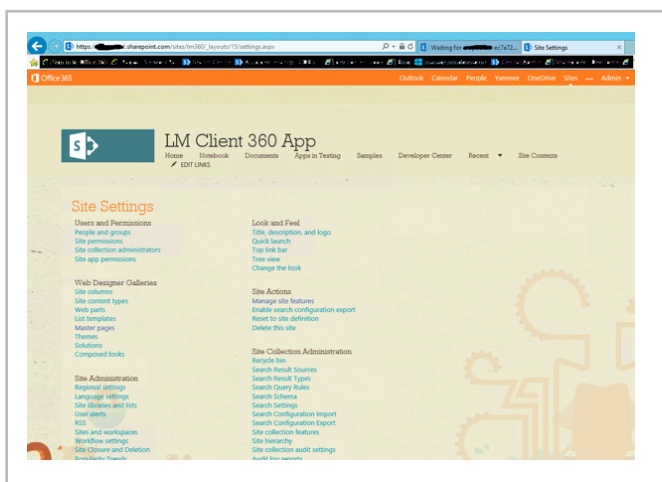


Imagen 3.- Tema aplicado a un sitio de SharePoint.

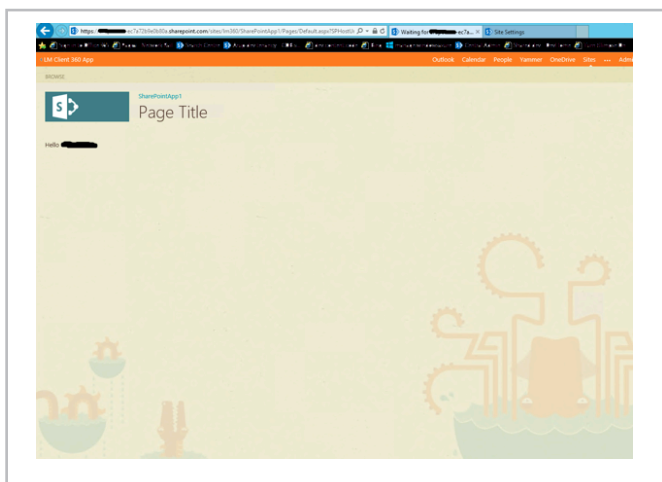


Imagen 4.- Tema aplicado a la SharePoint-Hosted App.

La página maestra aplicada, viene definida en cada página .aspx de la App en el atributo:

```
MasterPageFile="~masterurl/default.master"
```

Por defecto hace uso del token “~masterurl/default.master”, que se traduce por la URL de la galería de páginas maestras de la App, y la página maestra app.master, pero podemos cambiarlo y apuntar a la página maestra

que queramos. Llegado este punto, y viendo las páginas maestras que se provisionan por defecto, podríamos tener la tentación de usar, por ejemplo, la página maestra seattle.master, que nos proporcionaría la navegación, la caja de búsqueda, el acceso al a la página de configuración del sitio, etc. Sin embargo, aunque esto es a priori posible cambiando la página maestra:

```
MasterPageFile="~site/_catalogs/masterpage/seattle.master"
```

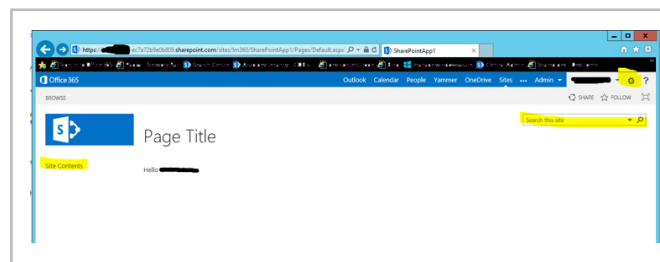


Imagen 5.- Página de una App usando seattle.master como página maestra.

Pronto nos daremos cuenta de que el resultado no es el esperado, ya que muchas de las redirecciones se intentaran hacer dentro de la App, y por lo tanto no funcionarán. La Imagen 6 muestra el resultado de pulsar sobre el enlace de “Contenidos del sitio” en la zona de navegación de la izquierda.

En algunos casos, puede que no sea suficiente con la app.master, y queramos utilizar nuestra página maestra personalizada, para añadir funcionalidad común a toda la App, como podría ser la navegación de la App. En este caso necesitaremos cambiar la página maestra por una página maestra que estará en la propia App y para ello, deberemos provisionar nuestra página maestra personalizada en el sitio de SharePoint creado para la App:

- Como sugerencia, recomiendo que se utilice la app.master como punto de partida a la hora de crear nuestra página maestra personalizada, ya que te dará la base para mantener la misma estética y tema que el sitio de SharePoint que hospeda la app. La app.master la puedes encontrar en: C:\Program Files\Common Files\microsoft shared\Web Server Extensions\15\TEMPLATE\GLOBAL
- Añadimos un nuevo módulo de SharePoint a la app, dicho modulo contiene nuestra master page personalizada.

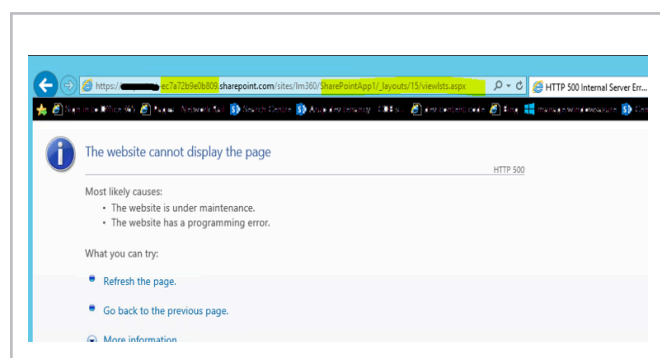
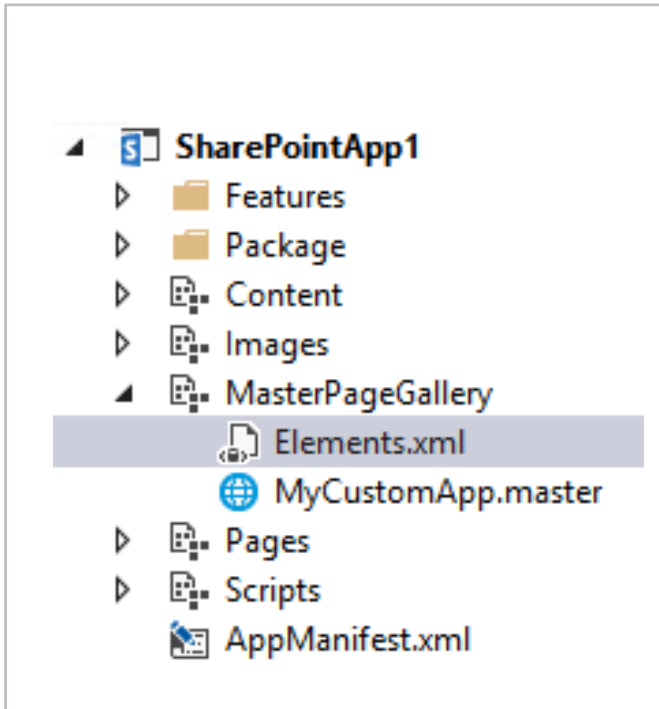


Imagen 6.- Error de redirección al pulsar “Contenidos del sitio”



- En las páginas de la App, editamos el atributo que referencia la master para que apunte a la página personalizada:

MasterPageFile=
 "~site/_catalogs/masterpage/MyCustomApp.master"

De esta forma, cuando se provisione la App, la página maestra estará en la galería de páginas maestras, y las páginas de la App harán uso de ella.

Imagen 7.- Módulo para provisionar la página maestra personalizada.

- Editamos el archivo Elements.xml y lo configuramos con el siguiente código:

LUIS MAÑEZ

MCPD SharePoint 2010 and Microsoft Active Professional

@luismanez

<http://geeks.ms/blogs/lmanez>



El autor destacado del Número 19

Ya tenemos al autor destacado del número 19 y el que completa a los cuatro del año.

En esta ocasión premiamos a Miguel Tabera, por su artículo “Buenas prácticas en infraestructura en SharePoint 2013”. Como siempre, gracias a Miguel y todos los autores por el esfuerzo en la revista.

El autor de año

Los siguientes artículos son los destacados de los números del último año:

- Número 16: Uso de Frameworks JavaScript en desarrollos SharePoint – Parte 1 por Adrián Díaz Cervera
- Número 17: Introducción a las plantillas de elementos de contenido (Display Templates) por Santiago Porras Rodríguez
- Número 18: Intranet Sociales: ¿El futuro? por Mariano Minolli
- Número 19: Buenas prácticas en infraestructura en SharePoint 2013 por Miguel Tabera y analizando las estadísticas de visitas vamos a premiar a Mariano Minolli por Intranet Sociales: ¿El futuro?

Muchas gracias a todos por mantener la calidad de los artículos porque gracias a ellos la revista sigue creciendo año a año.



Nosotros



Alberto Díaz

Alberto Díaz es SharePoint Team Lead en Encamina, liderando el desarrollo de software con tecnología Microsoft. Para la comunidad, ha fundado TenerifeDev (www.tenerifedev.com) con otros colaboradores, un grupo de usuarios de .NET en Tenerife, y coordinador de SUGES (Grupo de Usuarios de SharePoint de España, www.suges.es) y colaborador con otras comunidades de usuarios. Microsoft MVP de SharePoint Server desde el año 2011 y asiduo conferenciante en webcast y conferencias de tecnología de habla hispana.

Sitio Web: <http://blogs.encamina.com/negocios-sharepoint/>

Email: adiazcan@hotmail.com

Blogs: <http://geeks.ms/blogs/adiazmartin>

Twitter: [@adiazcan](https://twitter.com/adiazcan)



Fabián Imaz

Fabián Imaz, MVP de SharePoint Server trabaja en el mundo del desarrollo de software desde hace más de 10 años, teniendo la suerte de trabajar en distintas arquitecturas y tecnologías Microsoft. Pertenecer a la firma Siderys, <http://www.siderys.com> empresa de desarrollo de Software especializada en SharePoint 2007/2010/2013 y en desarrollo de soluciones inteligentes. Desde los comienzos Fabián ha trabajado en distintas comunidades donde organiza y promueve eventos locales para la difusión de tecnología dentro de los miembros de las mismas. Es director de la carrera SharePoint 2010 y SharePoint 2013 en Microsoft Virtual Academy, <http://www.mslatam.com/latam/technet/mva2/Home.aspx> y cuenta con un sitio en CodePlex con varios desarrollos <http://siderys.codeplex.com>.

Sitio Web: <http://www.siderysbsn.com>

Email: fabiani@siderys.com.uy

Blogs: <http://blog.siderys.com>

Twitter: [@fabianimaz](https://twitter.com/fabianimaz)



Gustavo Velez

Gustavo Velez es Ingeniero Mecánico y Electrónico; trabaja en el diseño e implementación de sistemas de IT basados en tecnologías de Microsoft, especialmente SharePoint, para Avanade (<http://www.avanade.com>), una compañía multinacional de IT. Propietario del sitio especializado en información sobre SharePoint en español <http://www.gavd.net> y autor de seis libros sobre SharePoint y sus tecnologías.

Sitio Web: <http://www.gavd.net>

Email: gustavo@gavd.net

Blogs: <http://geeks.ms/blogs/gvelez/>



Juan Carlos González Martín

Juan Carlos González, es Ingeniero de Telecomunicaciones por la Universidad de Valladolid y Diplomado en Ciencias Empresariales por la Universidad Oberta de Catalunya (UOC). Cuenta con más 10 años de experiencia en tecnologías y plataformas de Microsoft diversas (SQL Server, Visual Studio, .NET Framework, etc.), aunque su trabajo diario gira en torno a las plataformas SharePoint y Office 365. Juan Carlos es MVP de SharePoint Server desde el año 2008, coordinador del grupo de usuarios .NET de Cantabria (Nuberos.Net, www.nuberos.es) y co-fundador del Grupo de Usuarios de SharePoint de España (SUGES, www.suges.es), del Grupo de Usuarios de Cloud Computing de España (CLOUDES) y de la Comunidad de Office 365. Desde el año 2011 participa junto con Gustavo Vélez, Fabián Imaz y Alberto Díaz en la dirección de CompartiMOSS. Hasta la fecha, ha publicado cuatro libros sobre SharePoint y varios artículos en castellano y en inglés sobre la plataforma.

Email: jcgonzalezmartin1978@hotmail.com

Blogs: <http://geeks.ms/blogs/jcgonzalez> &

<http://jcgonzalezmartin.wordpress.com/>

¿Desea colaborar con CompartiMOSS?



La subsistencia del magazine depende de los aportes en contenido de todos. Por ser una revista dedicada a información sobre SharePoint en español, todo el contenido deberá ser directamente relacionado con Microsoft SharePoint y escrito en castellano. No hay limitaciones sobre el tipo de artículo o contenido, lo mismo que sobre el tipo de versión. Si desea publicar algo, por favor, utilice uno de los siguientes formatos:

- Artículos de fondo: tratan sobre un tema en profundidad. Normalmente entre 2000 y 3000 palabras y alrededor de 4 o 5 figuras. El tema puede ser puramente técnico, tanto de programación como sobre infraestructura, o sobre implementación o utilización.
- Artículos cortos: Máximo 1000 palabras y 1 o 2 figuras. Describen rápidamente una aplicación especial de SharePoint, o explica algún punto poco conocido o tratado. Experiencias de aplicación de SharePoint en empresas o instituciones puede ser un tipo de artículo ideal en esta categoría.
- Ideas, tips y trucos: Algunos cientos de palabras máximo. Experiencias sobre la utilización de SharePoint, problemas encontrados y como solucionarlos, ideas y trucos de utilización, etc.

Los formatos son para darle una idea sobre cómo organizar su información, y son una manera para que los editores le den forma al magazine, pero no son obligatorios. Los artículos deben ser enviados en formato Word (.doc o .docx) y las figuras por separado en un formato de alta resolución (.tif), todo comprimido en un archivo (.zip o .rar) con el nombre del autor y del artículo.

Si desea escribir un artículo de fondo o corto, preferiblemente envíe una proposición antes de escribirlo, indicando el tema, aproximada longitud y número de figuras. De esta manera evitaremos temas repetidos y permitirá planear el contenido de una forma efectiva.

Envíe sus proposiciones, artículos, ideas y comentarios a la siguiente dirección:

revista@compartimoss.com

adiazcan@hotmail.com

fabiani@siderys.com.uy

jcgonzalezmartin1978@hotmail.com

gustavo@gavd.net

| COMPARTIMOSS |