Nº38 diciembre 2018

CompartiMOSS

REVISTA ESPECIALIZADA EN TECNOLOGÍAS MICROSOFT

Entrevista Rodolfo Castro Aguilar

SharePoint Framework 1.7.0: ¿Qué hay de nuevo?

Azure Cognitive Services en un contenedor

SharePoint y Azure: Reconocimiento de imágenes

Contenido

Comparti MOSS

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 GENERAL

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

DISEÑO Y DIAGRAMACIÓN

- Santiago Porras Rodríguez

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

03

Editorial

Un nuevo número de la revista, un año más, y tanto Microsoft como CompartiMOSS siguen adelante. Aunque es tradición hacer un resumen del año que se acaba en la edición de diciembre, no es mucho lo que se puede decir en realidad al respecto. Solo algún slogan común, como, por ejemplo “negocios como de costumbre” podría describir el año en retrospectiva. Microsoft continua con su transición hacia “la nube”, teniendo el éxito de costumbre, tanto financiera como tecnológicamente; inclusive, la compañía termina el año siendo la compañía de tecnología mas valiosa del mundo, superando a todos sus rivales tradicionales.

Por nuestro lado, CompartiMOSS sigue su marcha, convirtiéndose cada vez mas y mas en el vocero de las tecnologías de Microsoft en castellano. Hace algunos años nos propusimos cambiar el objetivo de la revista de especializada en SharePoint a especializada en todas las tecnologías de Microsoft, y ahora podemos decir que el canje ha sido un éxito. Y, como en este numero podemos ver, la revista es un reflejo de lo que es Microsoft actualmente: mucho Azure y Office 365, algo de otras tecnologías, y muy poco de Windows.

Como de costumbre, el comité editorial de CompartiMOSS les desea un buen fin de año y un aún mejor próximo año. Y que disfruten de la revista tanto como nosotros, los autores y editores disfrutamos haciéndola.

El Equipo Editorial de CompartiMOSS

04

Trae los datos de Office 365 a tus aplicaciones con Microsoft Graph Data Connect

Estamos acostumbrados a crear aplicaciones centradas en personas que acceden a sus datos de Office 365 con Microsoft Graph API, pero ¿qué ocurre si queremos utilizar todos los datos de Office 365 para crear, por ejemplo, analíticas o incluso aplicaciones inteligentes? ¿Y si pudiésemos llevar toda esa información a Azure para poder trabajar con ella?

"Graph Data Connect es un servicio que nos permite obtener información de Office 365 a través de un pipeline de Azure Data Factory y copiarla en un contenedor de Azure"

Introducción a Microsoft Graph Data Connect

En el último Ignite, Microsoft anunció la disponibilidad de Microsoft Graph Data Connect en preview pública. Graph Data Connect es un servicio que nos permite obtener información de Office 365 a través de un pipeline de Azure Data Factory y copiarla en un contenedor de Azure para luego poder interactuar con ella.

Por ahora se puede obtener este conjunto de datasets:

DATA SET	DESCRIPCIÓN
BasicDataSet_v0.Contact_v0	Contiene información sobre los contactos de cada usuario. https://docs.microsoft.com/en-us/graph/api/resources/contact?view=graph-rest-1.0
BasicDataSet_v0.Event_v0	Contiene información sobre los eventos del calendario de cada usuario. https://docs.microsoft.com/en-us/graph/api/resources/event?view=graph-rest-1.0
BasicDataSet_v0.Message_v0	Contiene información sobre los mensajes del buzón de cada usuario. https://docs.microsoft.com/en-us/graph/api/resources/message?view=graph-rest-1.0
BasicDataSet_v0.SentItem_v0	Contiene información sobre los mensajes enviados por cada usuario. https://docs.microsoft.com/en-us/graph/api/resources/message?view=graph-rest-1.0
BasicDataSet_v0.User_v0	Contiene información de usuarios. https://docs.microsoft.com/en-us/graph/api/resources/user?view=graph-rest-1.0
BasicDataSet_v0.MailboxSettings_v0	Contiene información de la configuración del buzón de cada usuario https://docs.microsoft.com/es-es/graph/api/user-get-mailboxsettings?view=graph-rest-1.0
BasicDataSet_v0.MailFolder_v0	Contiene información sobre las carpetas del buzón de cada usuario https://docs.microsoft.com/en-us/graph/api/resources/mailfolder?view=graph-rest-1.0

Preparando el entorno

Antes de empezar a utilizar Graph Data Connect debemos realizar una serie de pasos para poder configurar el entorno:

1.– Crear un grupo de aprobadores de solicitudes de petición de datos: Todas las peticiones de datos de Office 365 a través de Graph Data Connect necesita que un usuario las apruebe, por lo que lo primero que tenemos que hacer es crear un grupo de aprobadores de estas solicitudes. Este grupo que creemos debe ser un grupo de seguridad habilitado para correo.

   Una vez creado el grupo tenemos que añadir los usuarios aprobadores de estas peticiones, por lo que añadiremos algún usuario a este grupo. Estos usuarios deben tener el rol de Administrador Global en Office 365 y deben tener activada la autenticación multifactor. En caso de no ser así, cuando se apruebe la solicitud, dará un error

2.– Activar Graph Data Connect en el tenant de Office 365: Para poder utilizar Graph Data Connect primero debemos activar el servicio. Para ello accedemos al Centro de Administración de Microsoft 365 (https://admin.microsoft.com) y dentro de Configuración -> Servicios y complementos selecciona Versión preliminar de la conexión de datos de Microsoft Graph

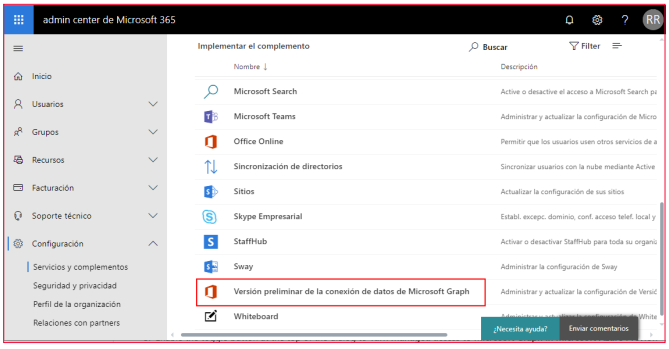

Imagen 1.- Acceso a la activación de Graph Data Connect.

Activa el servicio y selecciona el grupo de aprobadores que hemos creado en el paso anterior.

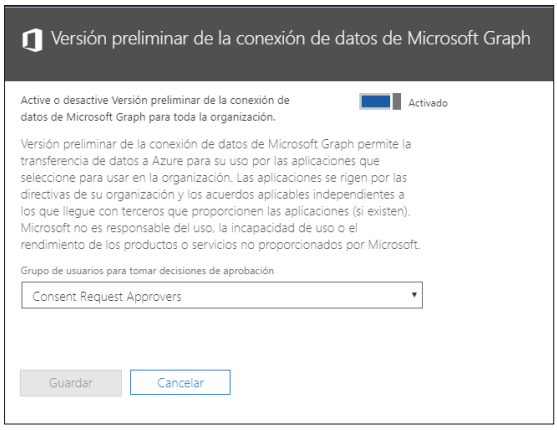

Imagen 2.- Activación de Microsoft Graph Data Connect.

Con estos pasos ya tendríamos configurado nuestro entorno para poder utilizar Graph Data Connect.

Extrayendo datos de Office 365 a un blob de Azure Storage

1.– Crear aplicación en Azure AD: Lo primero que tenemos que hacer es crear una aplicación en Azure AD que se utilizará como contexto de seguridad en la extracción de los datos. Para ello accederemos al portal de Azure (https://portal.azure.com) y dentro de Azure Active Directory -> Registro de aplicaciones seleccionamos Nuevo registro de aplicaciones

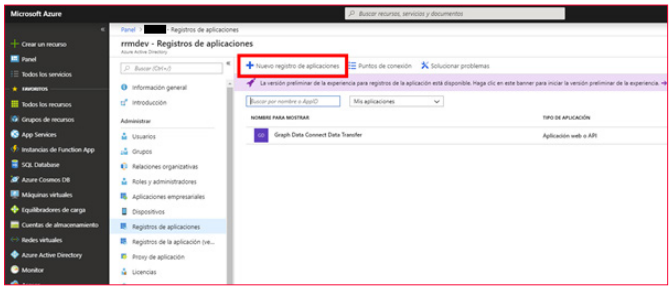

Imagen 3.- Creación de una aplicación en Azure AD.

En la siguiente pantalla pondremos un nombre a la aplicación, como tipo de aplicación seleccionamos Aplicación web o API y pondremos como url de inicio de sesión https://[tenant].onmicrosoft.com/[aplicacion].

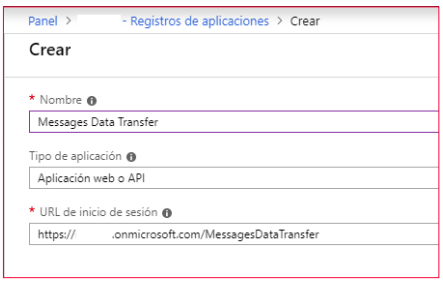

Imagen 4.- Datos de registro de la aplicación en Azure AD.

"Una vez creado el storage debemos dar permisos de contribución a la aplicación que hemos creado antes"

Copia el Id de aplicación, lo necesitaremos más adelante:

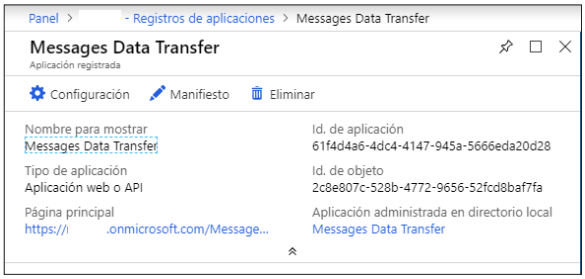

Imagen 5.- Id. De Aplicación.

Ahora necesitamos crear una clave para la aplicación, para ello dentro de la aplicación que hemos creado vamos a Configuración -> Claves y creamos una nueva clave. Copia la clave que se genera ya que sólo se muestra cuando se genera y después no podrás recuperarla.

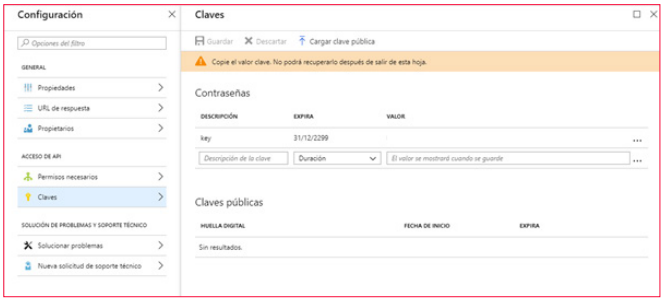

Imagen 6.- Generación de la clave para la aplicación.

Tienes que asegurarte que la aplicación tiene propietario, para ello vete a Propietarios y asegúrate que tiene un propietario y ese usuario es Administrador Global dentro de Office 365. Si no es así añádelo a la lista de propietarios.

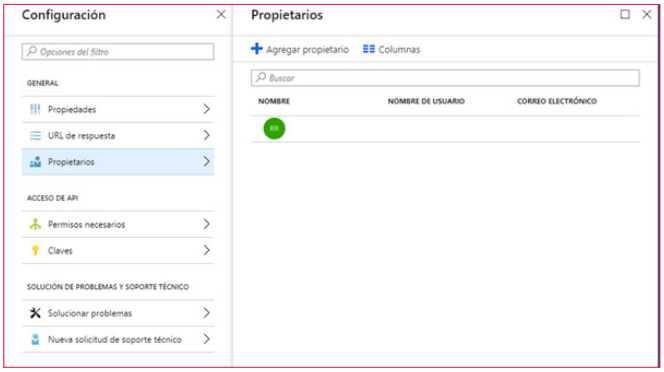

Imagen 7.- Configuración de los propietarios de la aplicación.

2.– Crear Azure Storage Blob: Ahora debemos crear un blob de Azure Storage para almacenar los datos que vamos a extraer de Office 365. Para ello accedemos al portal de Azure (https://portal.azure.com) y creamos un nuevo recurso de tipo Cuenta de almacenamiento: blob, archivo, tabla, cola. En la siguiente pantalla seleccionamos la suscripción, grupo de recursos y añadimos los siguientes valores:

- Nombre de la cuenta de almacenamiento: nombre único que pondremos al storage.

- Ubicación: Oeste de Europa.
- Rendimiento: Estándar.
- Tipo de cuenta: BlobStorage.
- Replicación: Almacenamiento con redundancia local (LRS).

Una vez creado el storage debemos dar permisos de contribución a la aplicación que hemos creado antes. Para ello dentro del storage que hemos creado vamos a Control de acceso (IAM) y Agregar asignación de roles

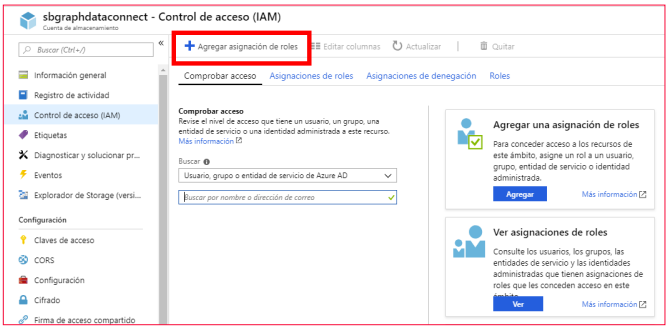

Imagen 8.- Asignación de roles al Blob Storage.

Añadimos los siguientes valores:

- Rol: Colaborador de datos de blobs de almacenamiento.
- Asignar acceso a: Usuario, grupo o entidad de servicio de Azure AD.
- Seleccionar: seleccionamos la aplicación que hemos creado en el paso anterior.

Ahora necesitamos un nuevo contenedor en el storage, para ello dentro de la cuenta de almacenamiento seleccionamos Blobs y añadir contenedor

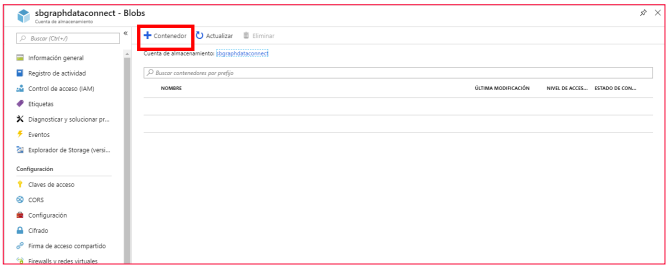

Imagen 9.- Creación del contenedor en el Storage.

Añadimos un nombre al contenedor y dentro de Nivel de acceso público seleccionamos Privada (sin acceso anónimo). Guardamos el contenedor y ya tendríamos preparado el contenedor para almacenar la información que vamos a obtener de Office 365.

3.– Crear Azure Data Factory Pipeline: Una vez que hemos creado la aplicación en Azure AD y el contenedor crearemos un pipeline en Azure Data Factory para extraer la información de Office 365.

Para ello dentro del portal de Azure creamos un nuevo Azure Data Factory, introducimos el nombre, suscripción, grupo de recursos, región y nos aseguramos de que sea de tipo Data Factory (V2). Una vez creado selecciona Crear y supervisar para acceder al editor de Azure Data Factory.

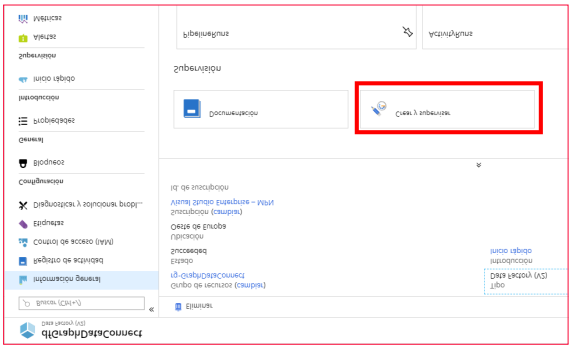

Imagen 10.- Creación del Azure Data Factory.

En el menú seleccionamos Author y creamos un nuevo pipeline.

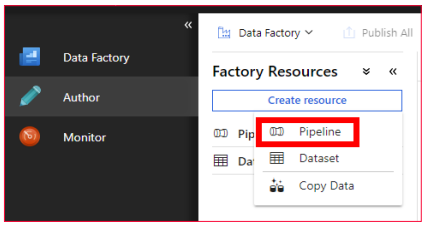

Imagen 11.- Creación del Pipeline.

Dentro del pipeline añadimos una nueva actividad Copy Data:

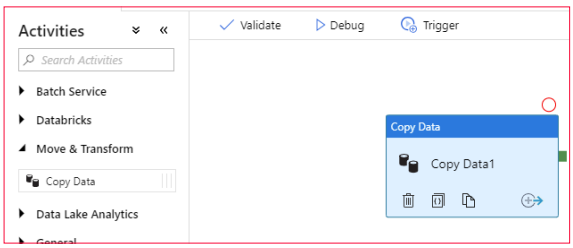

Imagen 12.- Añadiendo una actividad "Copy Data".

Ahora debemos configurar el origen y destino de la copia, para ello seleccionamos la actividad y en el panel inferior seleccionamos Source.

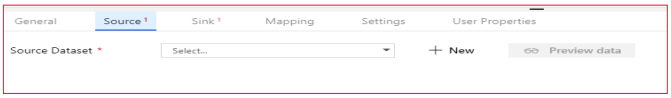

Imagen 13.- Configuración del origen y destino de la copia.

Pulsamos sobre el botón New para añadir un nuevo origen y seleccionamos Office 365 (Preview)

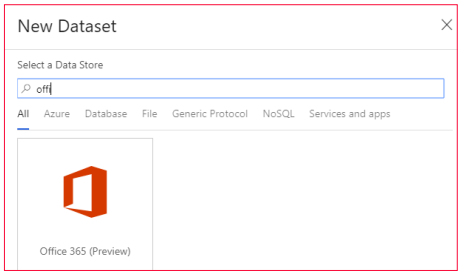

Imagen 13.- Selección del dataset de tipo Office 365 (Preview).

Al insertar se abrirá una nueva pestaña en la que podemos configurar el dataset. Dentro de esa pestaña, en el panel inferior, seleccionamos Connection y pul-

samos sobre New para añadir una nueva conexión.

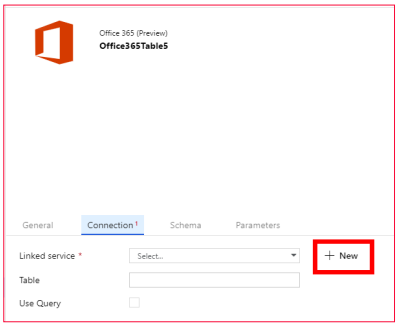

Imagen 14.- Acceso a la configuración del Dataset.

En esta pantalla es donde vamos a configurar la conexión para que utilice la aplicación de Azure AD que hemos creado anteriormente, por lo que añadiremos el Id de aplicación dentro de Service Principal ID y la clave que hemos generado en Service Principal Key.

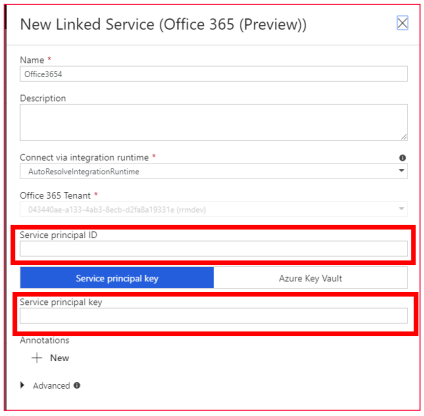

Imagen 15.- Parámetros de configuración del Dataset.

Una vez creada la conexi ón debemos seleccionar que datos queremos copiar, los seleccionaremos en el desplegable Table.

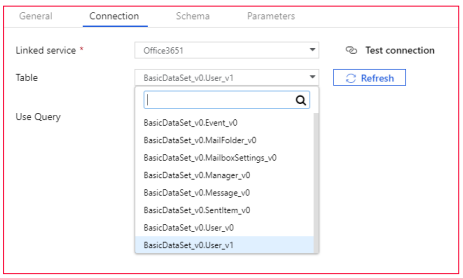

Imagen 16.- Selección de los datos a copiar.

Una vez creada la conexión vamos a la pestaña Schema y pulsamos sobre Import Schema. Se importará el schema de los datos que se van a obtener y es en esta pantalla donde podremos elegir qué información queremos sacar.

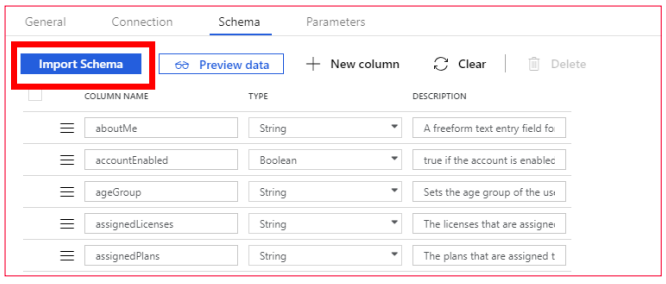

Imagen 17.- Importación del esquema.

Con esto ya tenemos configurado el origen de la copia, ahora nos queda configurar el destino. Volvemos a la pestaña del pipeline y seleccionamos Sink.

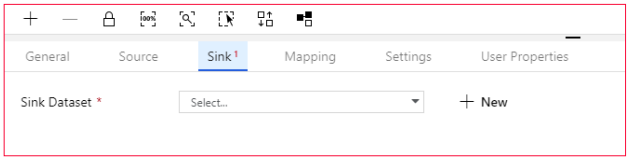

Imagen 18.- Configuración del destino.

Añadimos un nuevo destino y seleccionamos Azure Blob Storage. Al igual que cuando configuramos el origen, se abrirá una nueva pestaña para configurar la conexión al destino.

En el panel inferior añadimos un nuevo Linked Service.

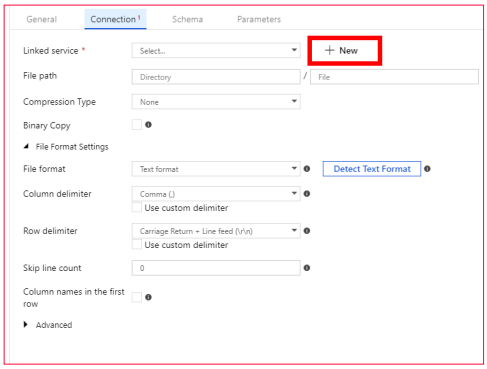

Imagen 19.- Añadiendo un nuevo servicio vinculado.

En esta pestaña configuramos la conexión. Para la conexión utilizaremos la aplicación de Azure AD que hemos creado antes por lo que seleccionaremos como método de autenticación Service Principal y añadiremos el Id de aplicación y la clave que hemos generado antes. También seleccionaremos la cuenta de almacenamiento que hemos generado en pasos anteriores.

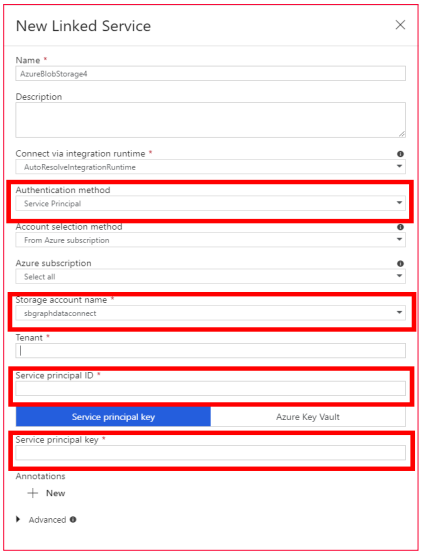

Imagen 20.- Parámetros de configuración del nuevo Linked Service.

Ahora debemos seleccionar el contenedor de destino para ello pulsamos Browse dentro de File Path y seleccionamos el contenedor que hemos creado antes.

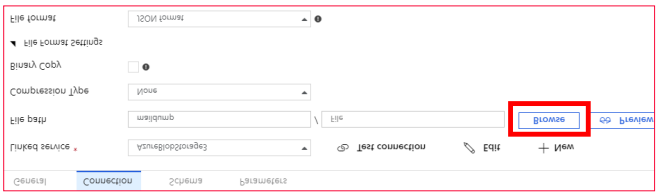

Imagen 21.- Selección del contenedor de destino.

Los datos que devuelve se envían en formato JSON, por lo que tendremos que seleccionamos JSON Format en el desplegable File Format y Set of objects en el desplegable File pattern.

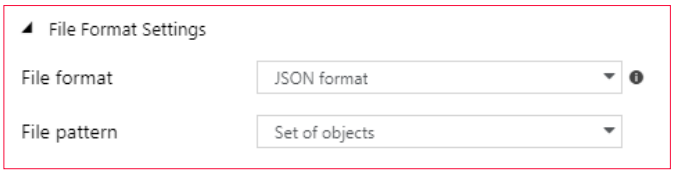

Imagen 22.- Selección del formato de los datos.

Una vez que hemos configurado el pipeline lo validamos y si todo es correcto lo publicamos

Imagen 23.- Validación del Pipeline y publicación.

4.– Ejecutar Azure Data Factory Pipeline: Ya tenemos creado el pipeline para copiar los datos, ahora vamos a ejecutarlo y probar que funciona correctamente. Para esto dentro de la pestaña del pipeline vamos a Trigger y seleccionamos Trigger Now

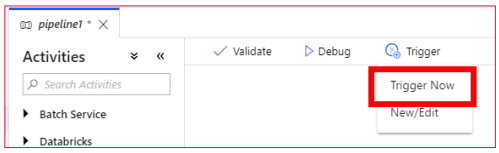

Imagen 24.- Ejecución del Azure Data Factory Pipeline.

"Una vez creado el storage debemos dar permisos de contribución a la aplicación que hemos creado antes"

Pulsamos sobre finalizar y comenzará la ejecución del pipeline. Dentro de la opción Monitor podremos ver la ejecución de las actividades del pipeline. La primera vez que ejecutamos el pipeline, pedirá consentimiento para obtener los datos. El proceso se quede en espera hasta que un usuario que pertenezca al grupo que hemos creado al principio autorice esa petición.

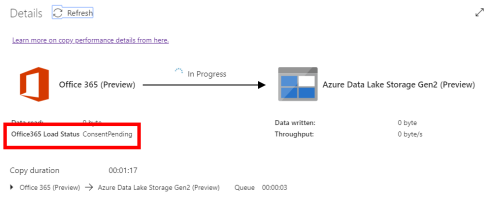

Imagen 25.- Monitorización de la ejecución del Azure Data Factory Pipeline.

En este punto se envía un e-mail al aprobador informando que existe una petición de acceso a datos que tiene que aprobar.

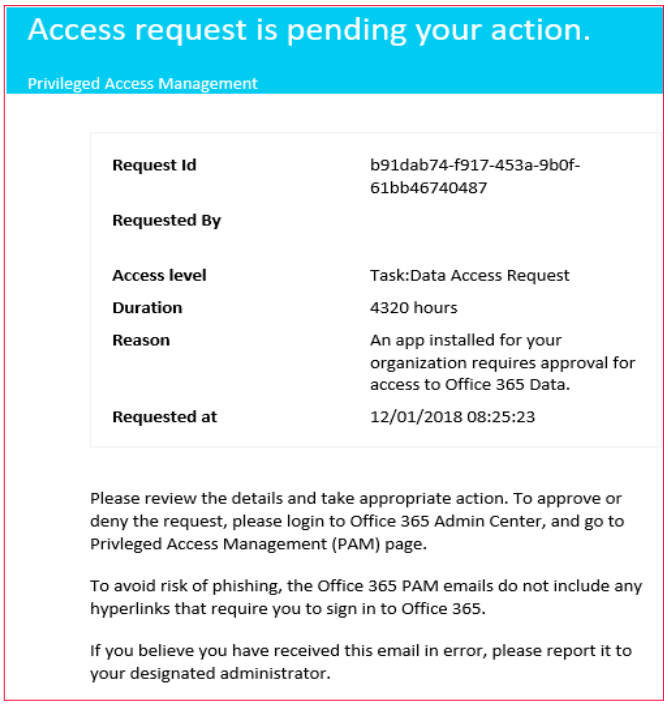

Imagen 26.- E-Mail enviado al aprobador.

Para aprobar o rechazar esta solicitud el usuario debe acceder a la página de Privileged Access Management (PAM). Para acceder tiene que ir a https://portal.office.com/adminportal/home#/Settings/PrivilegedAccess

En esta página se muestra todas las solicitudes realizadas. Seleccionamos la petición que nos interesa y aprobamos.

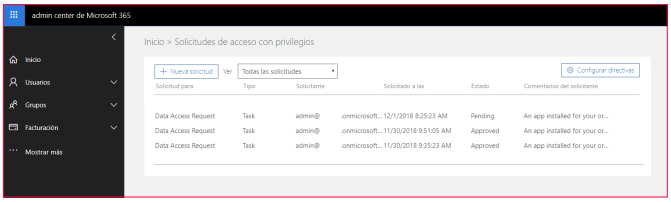

Imagen 27.- Listado de solicitudes realizadas en la configuración de PAM.

Una vez aprobada el flujo del pipeline continuará. Una vez que el pipeline termina la ejecución revisamos que los datos se han copiado correctamente. Para ello iremos al contenedor que hemos creado y vemos los datos que se han copiado.

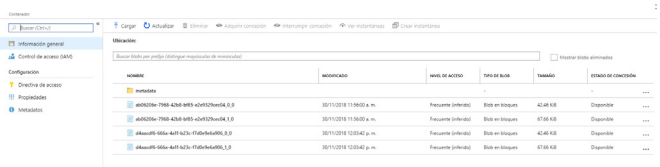

Imagen 28.- Visualización de los datos copiados.

Conclusión

Está claro que en Office 365 se almacena muchísima infor-

mación sobre sus usuarios, como se comunican, interactúan entre ellos, etc. Microsoft Graph Data Connect nos da la posibilidad de extraer esos datos y crear aplicaciones para explotarlos.

Aún está en fase de preview pública y necesita mejorar, sobre todo echo en falta poder sacar otra información además de los datasets que hoy en día están disponibles, pero esperemos que sigan añadiendo más.

Podéis encontrar más información dentro de su GitHub, https://github.com/OfficeDev/MS-Graph-Data-Connect

RUBÉN RAMOS MATEO
Technical Architect en Ricoh España
ruben_rm@outlook.com
@rubenr79

10

SharePoint Framework 1.7.0: ¿Qué hay de nuevo?

El pasado 8 de noviembre fue un día importante en el calendario SharePoint, ya que se liberó la esperada nueva versión del SharePoint Framework (SPFx), la 1.7.0, como ellos mismos dicen, probablemente la mayor release desde que apareció el SPFx.

Novedades incluidas en 1.7.0

Algunas features que han pasado a GA (Generally Available):

- Dynamic Data: también conocidos como WebParts conectados. Esta feature, que ya pudimos ver en la release anterior en forma de preview, permite conectar e intercambiar datos entre diferentes WebParts.
- Soporte para SharePoint Server 2019: Ahora podemos apuntar a SharePoint Server 2019 al crear el proyecto con el generador de Yeoman.
- React v16: por fin podemos trabajar con una versión más reciente de ReactJS, aunque todavía por detrás de la última versión desarrollada por los chicos de Facebook.

Además, las siguientes features se han agregado en forma de preview:

- SPFx como Tabs de Teams: Ahora podemos desarrollar soluciones SPFx, y ejecutarlas en Teams, como una Tab más. Esto es una gran mejora, ya que podemos extender Teams, y reutilizar nuestros skills de SPFx para ello.
- Apps basadas en Teams dentro de SharePoint: si estas creando provider-hosted apps para Teams, ahora también puedes usarlas en SharePoint (seria lo contrario al punto anterior, lo que demuestra que hay gran compatibilidad ahora entre Teams y SharePoint en cuanto a soluciones desarrolladas se refiere).
- Apps a página completa: este feature permite desarrollar páginas modernas donde el WebPart ocupa el cuerpo principal de la página.
- WebParts aislados: Otra de las peticiones mas esperadas era poder desarrollar WebParts que a nivel de permisos quedaran aislados unos de otros. En versiones anteriores, si un WebPart tenía permisos para llamar, por ejemplo, a Graph Api con permisos para leer el email del usuario, cualquier otro WebPart de la página tenia ese mismo permiso, lo necesitase o no.
- Suscripciones a Lista en el navegador: Ahora podemos obtener en nuestro componente, notificaciones “en tiempo real” cuando hay algún cambio a nivel de lista.
- Mejoras en el ciclo de vida de los ContentPlaceHolder: no hay mucho que destacar aquí. Básicamente Microsoft reconoce que no lo estaba haciendo muy bien en el ciclo de vida de los ContentPlaceHolder, asi que ha mejorado y re-hecho algunas de estas partes.

Creando proyectos (o actualizando) a 1.7.0

Para poder crear proyectos SPFx basados en el 1.7.0, primero debemos actualizar la versión del generador de Yeoman, ejecutando el siguiente comando:

```
npm install @microsoft/generator-sharepoint –global
```

Si lo que queremos es crear un proyecto nuevo, y utilizar las features en preview, acordaos de añadir la opción --plusbeta:

```
yo @microsoft/sharepoint --plusbeta
```

Si lo que queremos es actualizar un proyecto existente a la 1.7.0, recomendamos hacer uso de la herramienta Office365 CLI, con el comando Project-update (https://pnp.github.io/office365-cli/cmd/SPFx/project/project-upgrade/).

Este comando nos dará un reporte de todos los cambios que debemos aplicar, que, a groso modo, consisten en editar el fichero package.json y:

- Cambiar la versión de todos los paquetes de SPFx a 1.7.0.
- Actualizar la versión del paquete react y react-dom a la versión 16.3.2.
- Actualizar el paquete @types/react a v16.4.2.
- Actualizar el paquete @types/react-dom a v16.0.5.

Una vez realizados los cambios, se recomienda eliminar el directorio node_modules y volver a descargar todos los paquetes (bien sea con npm, yarn o el gestor que utilicéis).

Un par de notas no tan buenas

Conviene destacar que no todo son buenas noticias en cuanto a esta versión, ya que hay algunas cosas que esperábamos que se hubieran resuelto de otra manera, como,

por ejemplo:

- El generador de Yeoman hace preguntas sobre preview features (como Teams), incluso si no especificas la opción de usar las features en Preview.
- La versión de React, es la liberada en marzo 2018, por lo que es ya algo “vieja”.
- La versión de React Fabric está todavía en la v5, y esto es un poco decepcionante, ya que esta versión tiene bastantes issues conocidos que se solucionan en la v6.*.
- La versión de webpack también es algo antiguada, ya que sigue en la 3, y debería estar en la 4.*.
- Lo mismo con la versión de TypeScript, la 2.4.2 fue liberada en Julio 2017, por lo que llevamos mas de 1 año de retraso (actualmente TS esta en la v 3.1.6).

Aun así, no nos desanimemos, la 1.7.0 es una gran release, y teniendo en cuenta que ya estamos en la v16 de React, es muy posible que los updates ahora sean más rápidos y en breve estemos en versiones mas actuales.

Ejemplo de Teams-tab usando React

Pero basta de cháchara y vamos a ver un ejemplo de una de las features más esperadas de esta release, que es la posibilidad de desarrollar un WebPart y agregarlo a un Teams como una Tab más dentro de un Channel.

Como parte del repositorio en GitHub de SPFx, he creado un proyecto bastante completo que podéis descargar desde este link:

https://github.com/SharePoint/sp-dev-fx-WebParts/tree/master/samples/react-teams-tab-suggested-members

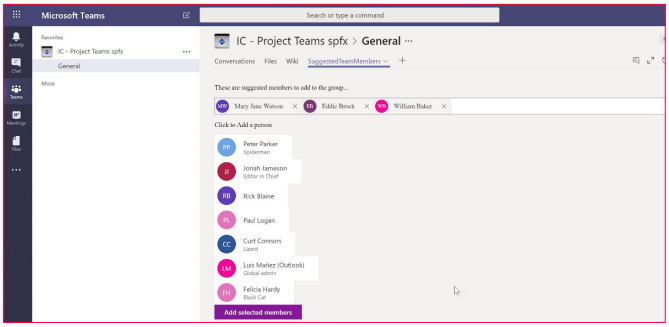

Imagen 1.- Ejemplo de Tab para Microsoft Teams creada con SPFx.

Para este artículo vamos a simplificarlo un poco, y explicaremos los fragmentos de código más importantes.

Partimos de un proyecto WebPart SPFx creado con la 1.7.0 y activando las features en preview.

Ahora disponemos de un namespace microsoftTeams, que ofrece un objeto Context, que nos va a ofrecer información del contexto de Teams, en el caso de que el WebPart se ejecute como una Tab de Teams. Así que lo primero en nuestro WebPart será instanciar ese contexto:

```
You, 7 days ago | 1 author (You)
export default class SuggestedTeamMembersWebPart extends BaseClientSideWebPart<ISuggestedTeamMemb

  private _teamsContext: microsoftTeams.Context;      You, 7 days ago • webpart added
```

Para cargar el contexto, tenemos que hacerlo en el onInit del WebPart. Quiza aquí, usando React, estéis pensando en hacer esto dentro del componentDidMount, pero a mí no me ha funcionado ahí, al parecer en ese punto del ciclo de vida del WebPart ya no podemos acceder al contexto de Teams, así que hay que cargar el contexto en el onInit, y pasarlo al componente de React en sus props.

```
protected onInit(): Promise<any> {
  let retVal: Promise<any> = Promise.resolve();
  if (this.context.microsoftTeams) {
    retVal = new Promise((resolve, reject) => {
      this.context.microsoftTeams.getContext(context => {
        this._teamsContext = context;
        resolve();
      });
    });
  }
  return retVal;
}
```

En principio, hasta que no se resuelve la Promise del evento OnInit, no se ejecutara el render del WebPart, por lo que en ese momento podemos pasar el contexto al componente “hijo” del WebPart (el componente React).

```
public render(): void {
  const element: React.ReactElement<ISuggestedTeamMembersProps > = React.createElement(
    SuggestedTeamMembers,
    {
      teamsContext: this._teamsContext,
      graphHttpClient: this.context.graphHttpClient,
      groupId: this.context.pageContext.site.group.id
    }
  );

  ReactDom.render(element, this.domElement);
}
```

Llegado a este punto, en el Render del componente de React, podemos acceder al contexto de Teams con:

this.props.teamsContext

Y acceder a diferentes propiedades como:

this.props.teamsContext.teamId
this.props.teamsContext.teamName
this.props.teamsContext.channelId
this.props.teamsContext.channelName

Fijaros en la potencia de esto, ya que conociendo estas propiedades del Team, podemos acceder a la Graph API (usando el conocido graphHttpClient), y poder extender la funcionalidad de Teams.

Conclusión

Como hemos comentado durante el artículo, estamos ante una de las actualizaciones mas esperadas, y que abre un importante abanico de posibilidades de desarrollo y extensión, no ya sólo de SharePoint, sino ahora también de Teams. Animaros a probarlo. Por nuestra parte, en los siguientes números iremos viendo con más detalle todas las novedades del 1.7.0.

LUIS MAÑEZ
SharePoint/Cloud Solutions Architect en ClearPeople LTD
@luismanez
https://medium.com/inherits-cloud

12

Hub Sites en Office 365, acceso vía REST

Nadie puede negar que el crecimiento de Office 365 en los últimos años ha sido exponencial y cada vez son más las organizaciones, públicas, privadas y educativas que están adoptando la plataforma colaborativa como la solución integral para el manejo de información y procesos. También es bien sabido que cuanto más grande es la organización, la gobernabilidad se hace una tarea titánica e incluso a los usuarios no se les hace nada fácil acceder a la información que necesitan debido a que no conocen en que sitio/área/departamento, o como se llamé dentro de la organización a la que pertenezca, se encuentra guardada. Bueno para facilitar la tarea de brindar acceso a los sitios, es que llegaron los Hub Sites.

¿Qué son? Básicamente son concentradores de sitios, es decir, permiten crear un sitio y configurarlo para que sea por así decirlo la página de llegada de un conjunto de sitios. Imaginen una organización determinada por gerencias y cada gerencia cuenta con una determinada cantidad de departamentos y cada departamento con secciones. Una primera solución sería armar una colección de sitios y dentro de la misma, ir armando un árbol de subsitios, así hasta lograr configurar la estructura organizacional. Pero está solución tiene un problema: que pasaría si mañana un departamento de una gerencia es movido a otra gerencia o si una gerencia es partida en dos.

La recomendación en Office 365, es evitar tener muchas anidaciones de sitios en una misma colección de sitios, y crear tantas colecciones de sitios como sean necesarias. Esto permite tener una gobernabilidad más descentralizada, incluso permite delegar la misma sin tener que andar haciendo grandes configuraciones de permisos y brindar así responsabilidades claras a los usuarios. Siguiendo con esta línea, se crearían una colección de sitios para gerencia, para cada departamento e incluso se podría llegar a pensar en que cada sección fuera también una colección de sitios dentro de nuestro Office 365, pudiendo así tener sitios raíz de colaboración moderna o comunicación.

"son concentradores de sitios, es decir, permiten crear un sitio y configurarlo para que sea por así"

En Office 365 no existe aún una opción o plantilla para crear un sitio del tipo Hub Sites e incluso no es algo que sea estrictamente necesario, debido a que la configuración se podría realizar por única vez y no volver a necesitarla. Cualquier sitio de comunicación o colaboración (experiencia moderna) puede ser configurado como un Hub Site dentro de Office 365. Entonces para poder armar la solución descrita anteriormente lo que se debería hacer primero es crear todas las colecciones de sitios que se necesitan utilizando algunas de las plantillas mencionadas. Una vez creada la estructura de colecciones de sitios, el próximo paso es crear nuestros concentradores de sitios (Hub Sites), para lo cual se utilizará un comando PowerShell que lo que hace es indicar que el sitio se transformará en un Hub Sites dentro de Office 365, pero que además se podrá seguir usando como sitio.

Ejecute la siguiente línea de PowerShell conectado a Office 365:

```
Register-SPOHubSite URL or Site ID
```

Al comando debe proporcionarle la URL de la colección de sitios o bien el ID (GUID) para que se ejecute correctamente, en el siguiente enlace puede acceder a más información del comando: https://docs.microsoft.com/es-mx/sharepoint/create-hub-site.

El próximo paso es configurar en cada sitio que pertenece a la Hub Site. Lo bueno es que para esta configuración no se debe ejecutar ningún comando y se pude hacer desde el propicio sitio. Acceda a una colección de sitios, seleccione la opción de "Configuración del Sitio" y luego seleccione la opción "Información del Sitio", donde se puede ver una opción de configuración para seleccionar un Hub Site disponible para establecer donde se quiere que la colección de sitios sea mostrada.

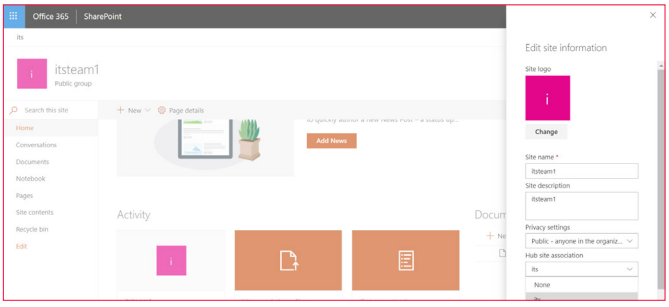

Imagen 1. – Configuración Hub Site en una colección de sitios.

> ***"es evitar tener muchas anidaciones de sitios en una misma colección de sitios si no más crear tantas colecciones de sitios como sean necesarias"***

Una vez que los sitios fueron configurados, en cada Hub Site se dispone de una WebPart llamada Sitios (Sites) la cual, al agregarla a una página, nos permite listar todos los sitios asociados con dicho Hub Site y brindar así un acceso rápido. Los permisos se tienen en cuenta, es decir que, si un usuario no tiene permisos para ver un sitio, este no se le mostrara en el Hub Site. Ahora, siguiendo con nuestro ejemplo, se puede ver lo fácil que es armar la estructura organizacional planteada al principio y bridar una navegación sencilla a los usuarios como se puede ver en la siguiente imagen.

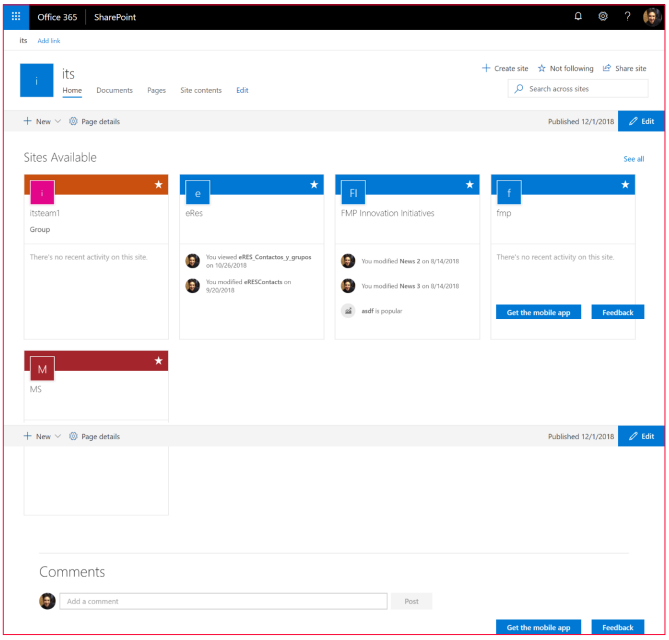

Imagen 2.- Hub Site en Office 365.

Como se ha podido ver, está solución de diseño utilizando Hub Site es sencilla y fácil de implementar. Pero también permite acceder a las colecciones de sitios de forma programática a través de un punto de acceso REST recientemente liberado en Office 365. Esto, que parce algo natural para cualquier programador, en el pasado no era tan sencillo, ya que para poder acceder al listado de una colección de sitios en Office 365 programáticamente se requería contar con permisos de administración y solo se podía a través de PowerShell o una librería de clases en CSOM. Bueno, esto si bien no brinda acceso a la lista completa de colecciones de sitios en Office 365, si deja acceder a todas las colecciones de sitios asociadas a un Hub Site determinado, lo cual permite construir componentes que accedan a información más allá de la colección de sitios donde se están ejecutando.

Se debe tener en cuenta que estos puntos de acceso en REST se encuentran en BETA y no se deben usar en desarrollos que van a ser puestos en producción antes que sean liberados debido a que pueden tener cambios y esto afectar el componente que se esté realizando.

El primer punto de acceso se llama "HubSites" y se puede consultar utilizando el verbo GET de REST, como toda consulta es relativa y la URL al punto de acceso es la siguiente _api/HubSites y nos devolverá el lista de sitios que fueron creados como HubSites, en el siguiente enlace se puede acceder a más información del punto de acceso REST https://docs.microsoft.com/es-mx/sharepoint/dev/features/hub-site/rest-hubsites-method

El otro punto de acceso interesante se llama "HubSiteData" y este devuelve información relativa a un HubSite determinado, la URL al punto de acceso es un poco diferente _api/web/HubSiteData y también se utiliza el verbo GET para poder acceder, en el siguiente enlace podrán tener más información con respecto al este punto de acceso https://docs.microsoft.com/es-mx/sharepoint/dev/features/hub-site/rest-hubsitedata-method

Ahora, si combinamos estos dos puntos de acceso, con el del motor de búsqueda, _api/search se podrá construir una consulta en el sitio hub que se desea y poder acceder al listado de sitios que el Hub Site tiene asociado. A modo de ejemplo la siguiente consulta devuelve el listado de sitios asociado a un un Hub Site:

/_api/search/query?querytext='DepartmentId:{be-f2fc44-6c1a-4dc2-ba96-fdfd4424cecd} contentclass:STS_Site'&selectproperties='Title,Path,DepartmentId,SiteId'

> ***"siguiendo con nuestro ejemplo, se puede ver lo fácil que es armar la estructura organizacional planteada al principio"***

La clave en esta consulta está en la propiedad DepartamentId, la cual almacena el ID del sitio Hub Site asociado, esto nos permite hacer la consulta pidiendo todos los sitios donde el valor de dicha propiedad sea la del Hub Site que se desea.

Conclusiones

En este artículo se trató de demostrar como utilizando la característica de HubSites se puede construir estructura organización de forma rápida y sencilla. Pero también se vio que utilizando los nuevos puntos de acceso REST se pueden crear componentes transversales a la organización para que los usuarios accedan a la información.

FABIÁN IMAZ
Office Apps and Services MVP
fabiani@siderys.com.uy
@fabianimaz

14 Entrevista Rodolfo Castro Aguilar

Mi nombre es Rodolfo Castro Aguilar, nací y crecí en la Ciudad de México, he tenido la fortuna de estar en Los Ángeles, California y una corta estadía en Medellín, Colombia., durante ya casi 10 años he estado trabajando con la parte de Comunicaciones Unificadas de Microsoft, ya desde la versión OCS 2007 he visto cómo cambia y evoluciona hasta lo que tenemos hoy en día con Skype4B y Teams.

He tenido la fortuna de ser galardonado como MVP en los años 2014 (Lync) ,2015 (Skype4b), 2016 (Office Server and Services) y 2018 (Office Apps and Services).

Actualmente estoy laborando en Polycom, como Microsoft UC Consultant para CALA.

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

Empecé de pequeño, tengo un tío que estaba en la universidad en ese momento, y cuando yo no tenía clases iba con él, me dejaba en el centro de cómputo jugando y fue desde esa etapa que fui adentrándome al mundo de la tecnología, seguido a que cuando uno entra en Educación Secundaria aquí en México nos hacen elegir un taller para desarrollarnos, obviamente escogí computación. Computadoras con disquetes de 5 ¼, programación con QBASIC fueron lo que termino de decidir qué era lo que yo quería ser de grande. Encaminado en ello entro a la Preparatoria de la Universidad La Salle, para prepararme para después tomar mi carrera en Ingeniería Cibernética y Sistemas Computacionales. Y pues lo demás se cuenta solo, aquí me tienes.

¿Cuáles son tus principales actividades tecnológicas hoy en día?

Actualmente estoy en un periodo bastante grato para mí, ya que como les he contado mi foco es Comunicaciones de Microsoft, ahora con el cambio de Skype a Teams me ha tocado mantenerme actualizado en todo este ambiente, desde las fases y mejores prácticas para la migración, la actualización de componentes y devices (también trabajando para Polycom me toca revisar esto) Mantenerme al pendiente de los nuevos anuncios que hace Microsoft, de las nuevas características que va liberando para llevarnos todo a Teams, o la integración y coexistencia que puede haber.

También algo que me gusta es automatizar todo para mi trabajo, así que siempre ando trabajando en algún script que me facilite las cosas y compartirlo en un futuro con la comunidad.

Y ya en mi trabajo, en tiempos que no tengo proyecto asignado me pongo a hacer pruebas con versiones betas para encontrar algún fallo o algunos tutoriales o webinars para la comunidad.

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

Como buen Ingeniero el tiempo que no paso laborando trato de estar con mi familia, tengo dos hermosos hijos, el niño de 6 y la niña de casi 2 años. Que absorben una buena parte de ese tiempo, son buenos momentos. Mi esposa, que tratamos de ver series una vez que los pequeños están descansando. Y de vez en cuando salimos al cine.

Me gusta jugar Xbox con mi hijo, a él le fascina matar zombis. Cada 15 días, voy al estadio a ver a mi equipo de futbol jugar en el Estadio Azteca.

Nada extraordinario, pero todo lo que vivo son muy buenos recuerdos que no cambiaría por nada del mundo.

También estoy enfocado a la parte de Comunidades, tengo mi propia comunidad de Skype y Teams, y estamos por organizar una conferencia anual, siendo esta la primera en su tipo en Latam, esperando que sea todo un éxito y la podamos realizar año con año como se tiene planeado.

Además, tengo un proyecto con un Puppet o Marioneta, llamado Puppets 365, el cual quiero enfocarme a enseñar al público en general a utilizar la suite de Office 365.

¿Cuáles son tus hobbies?

Colecciono cualquier cosa que tenga que ver con Batman, son un fanático completamente de ese personaje, tengo desde playeras, sudaderas, chamarras, y una colección grande de Funko Pop de solamente ese personaje. Mis familiares tratan de obsequiarme plumas, tazas, llaveros, estampas, sobre todo mi esposa. No le fascina que tenga tantas cosas, pero me consciente y me da más cosas para coleccionar.

Actualmente voy empezando una colección de calcetas de personajes de caricaturas y videojuegos.

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

Cloud, Cloud, Cloud y más Cloud. La tendencia a irnos a la nube ya es una necesidad, más que una visión. Ahora nosotros como Consultores tenemos que desarrollar más nuestros soft skills y no solo saber la parte tecnológica, sino entender bien las necesidades del cliente, entender que es lo que el cliente ocupa y con base en ello proponerles la mejor solución. En la parte de comunicaciones que me toca ver a mí, aún hay algunas limitaciones tecnológicas en LATAM que obligan a tener implementaciones hibridas, pero con el paso del tiempo no veo complicado que tengamos todo en la nube incluyendo la parte de telefonía que es lo que más trabajo cuesta en este momento.

Apoyando todo esto, veo que la tecnología ira desarrollando más el uso de home office para varias actividades y la integración de varias soluciones en un mismo cliente (ya se está dando en Teams) un mayor uso de bots con IA, todo utilizando la nube.

RODOLFO CASTRO AGUILAR
MVP Office Apps and Services
Twitter : @ucblogmx
facebook.com/groups/SkypeTeamsUG/
ucblogmx.com

16

SharePoint y Azure: Reconocimiento de imágenes

El servicio de Reconocimiento de imágenes de Azure forma parte del grupo de Servicios de Conocimiento (Cognitive Services), bajo los servicios de Visión Computarizada (Computer Vision). El servicio ofrece ayuda con diferentes tareas:

- Análisis de imágenes: Esta función devuelve información sobre el contenido visual que se encuentra en una imagen, ofreciendo etiquetado, descripcion en cuatro idiomas para identificar el contenido, ayuda a detectar contenido potencialmente para solo adultos, tipos de imágenes y esquemas de color.
- Reconocimiento de celebridades y sitios importantes: reconoce imágenes de aproximadamente 200.000 personalidades mundiales y 9.000 sitios importantes. Utiliza el servicio de "Análisis de imágenes" descrito en el punto anterior.
- OCR: Detección de texto en una imagen usando Optical Character Recognition (OCR). Vea el articulo "SharePoint y Azure: Reconocimiento Óptico de Caracteres (OCR)" en CompartiMOSS número 37 de septiembre 2018 (http://www.compartimoss.com/revistas/numero-37/sharepoint-y-azure-ocr).
- Reconocimiento de texto manual: Similar al servicio de OCR, pero para texto escrito a mano en una imagen.
- Análisis de video: analiza frames de videos en tiempo (casi) real y los envía a otro servicio para presentación y análisis.
- Generación de thumbnails: modifica el tamaño y estilo de imágenes.

Azure Image Recognition y SharePoint

El servicio de Reconocimiento de imágenes del Azure Computer Vision API se puede utilizar en SharePoint para enriquecer automáticamente la información en el sistema. Cuando se suben imágenes a SharePoint, en realidad el sistema no puede indexar su contenido, solamente los metadatos que posee. Por medio de este servicio de Azure es posible encontrar información sobre la imagen, y agregarla al sistema como metadatos, permitiendo clasificarla con exactitud, y obtener resultados de búsqueda más precisos.

En el ejemplo que se va a desarrollar enseguida se utilizan los servicios del Computer Vision API para extraer información de una imagen de un personaje y agregarla a varios campos en la Librería a donde se sube, además de crear un thumbnails. La Biblioteca de SharePoint dispone para esto de campos de texto para insertar el nombre del personaje, su edad estimada en la foto, si la imagen es identificada como contenido para adultos, y un campo de imagen para el thumbnail.

Configuración del Azure Computer Vision API

Para utilizar el Computer Vision API es necesario crear primero el servicio en Azure. Para acceder a la página de administración de Azure (https://portal.azure.com) necesita sus credenciales (es posible obtener una cuenta temporal de prueba de Azure desde la página de Microsoft https://azure.microsoft.com/en-us/try/cognitive-services/).

Para crear un servicio (de pago) en Azure:

1.– Entre al portal de manejo de Azure (https://portal.azure.com) utilizando sus credenciales.

2.– Vaya a la sección de "Resource Groups" y cree un nuevo Grupo de Recursos (también es posible reutilizar un grupo ya existente).

3.– Cree un servicio de "Computer Vision API":

- En el Resource Group, utilice el botón de "+Add" para crear un recurso, busque por "computer vision" en la casilla de búsqueda y seleccione "Computer Vision" en los resultados.
- Asígnele un nombre al servicio y utilice el Grupo de Recursos deseado. En la casilla de "Pricing tier" seleccione un nivel dependiendo de la cantidad de consultas a esperar por segundo, lo que determina el precio del servicio (por mil consultas).

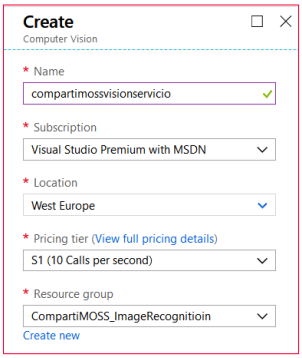

Imagen 1.- Creación del servicio de Computer Vision API.

4.– Una vez creado el servicio, haga clic sobre su nombre en la lista de recursos del Resource Group, vaya a "Keys" y copie el valor de "Key 1".

Utilizando el Azure Computer Vision con SharePoint

Las llamadas al servicio de Azure se realizan por medio de REST o por medio del Computer Visio API SDK. En el siguiente ejemplo, que utiliza ambas formas de uso del servicio, se va a subir una imagen a una Biblioteca de SharePoint y luego, utilizando un WebHook, hacer que una Función de Azure haga dos llamadas al servicio:

- La primera llamada es para utilizar el servicio de Análisis, el que devuelve varias propiedades de la imagen: si es una imagen de una personalidad o no, si el contenido es para adultos o no, la edad estimada de la persona (y varias otras propiedades que no se van a utilizar en el ejemplo, como el tamaño de la imagen, tipo de imagen, etc.). Utiliza una llamada REST.
- La segunda llamada es al servicio de generación de thumbnails, el que devuelve otra imagen de dimensiones especificadas. Utiliza una llamada del Computer Visio API SDK.

Nota: la creación y configuración de una Función de Azure se puede encontrar el en artículo "SharePoint y Azure – Azure Functions" (http://www.compartimoss.com/revistas/numero-30/sharepoint-y-azure-azure-functions). La configuración y utilización de WebHooks de SharePoint se puede encontrar en el artículo "Eventos sobre SharePoint Online con Webhooks" (http://www.compartimoss.com/revistas/numero-32/eventos-sobre-sharepoint-online-con-webhooks).

1.– Cree una cuenta de Funciones básica en el Grupo de Recursos, asignándole un nombre, Plan de Servicios y cuenta de Azure Storage

2.– Utilizando Visual Studio 2017 (o Visual Studio 2015 con el AddIn para programar Funciones de Azure), cree una nueva solución del tipo "Azure Functions". Una vez creada la solución, agréguele una Función del tipo "Http Trigger" (utilizando "Azure Functions v2") con derechos de acceso anónimos y cambie el nombre de la Función una vez creada (por defecto se llamará "Function1").

3.– Agréguele a la solución los paquetes NuGet "AppForSharePointOnlineWebToolkit", " Microsoft.Azure.CognitiveServices.Vision.ComputerVision" y "Newtonsoft.Json".

4.– Reemplace toda la rutina "Run" con el siguiente código:

"Las llamadas al servicio de Azure se realizan por medio de REST o por medio del Computer Visio API SDK"

```
[FunctionName("ImageAnalyze")]
public static async Task<HttpResponseMessage> Run([HttpTrigger(AuthorizationLevel.Anonymous, "post", Route = null)]
HttpRequestMessage req, TraceWriter log)
{
    log.Info("*** ImageAnalyze function processed a request ***");

  // Registration
      string validationToken = HelpFunctions.GetValidationToken(req);
  if (validationToken != null)
  {
    log.Info($"---- Processing Registration");
     var myResponse = req.CreateResponse(HttpStatusCode.OK);
       myResponse.Content = new StringContent(validationToken);
    return myResponse;
  }

  // Changes
  var myContent = await req.Content.ReadAsStringAsync();
    var allNotifications = JsonConvert.DeserializeObject<ResponseModel<NotificationModel>>(myContent).Value;

  if (allNotifications.Count > 0)
  {
    log.Info($"---- Processing Notifications");
     string siteUrl = ConfigurationManager.AppSettings["whSiteListUrl"];
    foreach (var oneNotification in allNotifications)
    {
      // Login in SharePoint
       ClientContext SPClientContext = HelpFunctions.LoginSharePoint(siteUrl);

      // Get the Changes
      GetChanges(SPClientContext, oneNotification.Resource, log);
    }
  }

  return new HttpResponseMessage(HttpStatusCode.OK);
}
```

Esta rutina primero se encarga de hacer el registro del WebHook (si la consulta contiene un parámetro "validationtoken" en el Query String) utilizando la rutina "GetValidationToken":

```
public static string GetValidationToken(HttpRequestMessage req)
{
  string strReturn = string.Empty;

  strReturn = req.GetQueryNameValuePairs()
        .FirstOrDefault(q => string.Compare(q.Key, "validationtoken", true) == 0)
    .Value;

  return strReturn;
}
```

Después de registrado el WebHook, cada consulta es procesada para extraer las notificaciones que contiene. En cada notificación de la colección de notificaciones se hace un logeo en SharePoint para obtener los cambios detectados en la Lista (por medio de la rutina "GetChanges"). En la variable "whSiteListUrl" del App Settings de la función se encuentra el URL del sitio en donde está la Lista a examinar ("https://[Dominio].sharepoint.com/sites/[NombreSitio").

5.– La rutina “GetChanges” recibe el contexto de SharePoint y el identificador de la Lista, y tiene la forma:

```
static void GetChanges(ClientContext SPClientContext, string ListId, TraceWriter log)
{
  // Get the Images List
  Web spWeb = SPClientContext.Web; //.Site.RootWeb;
  List myList = spWeb.Lists.GetByTitle(ConfigurationManager.AppSettings[“whListNameImageAnalyze”]);
  SPClientContext.Load(myList);
  SPClientContext.ExecuteQuery();

  // Get the Thumbnail List
  Web spWebThumb = SPClientContext.Web;
    List myListThumb = spWebThumb.Lists.GetByTitle(ConfigurationManager.AppSettings[“whListNameImageAnalyze-Thumbails”]);
  SPClientContext.Load(myListThumb);
  SPClientContext.ExecuteQuery();

  // Create the ChangeToken and Change Query
  ChangeQuery myChangeQuery = HelpFunctions.GetChangeQueryNew(ListId);

  // Get all the Changes
  var allChanges = myList.GetChanges(myChangeQuery);
  SPClientContext.Load(allChanges);
  SPClientContext.ExecuteQuery();

  foreach (Change oneChange in allChanges)
  {
    if (oneChange is ChangeItem)
    {
      int myItemId = (oneChange as ChangeItem).ItemId;

      // Get what is changed
      log.Info($”---- Changed ItemId : “ + myItemId);
      ListItem myItem = myList.GetItemById(myItemId);
      Microsoft.SharePoint.Client.File myFile = myItem.File;
       ClientResult<System.IO.Stream> myFileStream = myFile.OpenBinaryStream();
      SPClientContext.Load(myFile);
      SPClientContext.ExecuteQuery();

      // The picture as Byte Array
       byte[] myFileBytes = CognitiveRoutines.ConvertStreamToByteArray(myFileStream);

      // Analyze the picture
     ImageAnalyzeResult myResult = CognitiveRoutines.GetAzureImageAnalyze(myFileBytes).Result;
        log.Info($”---- Image Analyze Result : “ + JsonConvert.SerializeObject(myResult));

      // Create the Thumbnail
       Stream myThumbnail = CognitiveRoutines.GetAzureImageThumbnail(myFileBytes).Result;
      log.Info($”---- Image Thumbnail created”);

      // Upload the Thumbnail
      FileCreationInformation infoThumb = new FileCreationInformation();
      infoThumb.Content = CognitiveRoutines.ConvertStreamToByteArray(myThumbnail);
     infoThumb.Url = myListThumb.ParentWebUrl + “/” + myListThumb.Title + “/thumb_” + myFile.Name;
       Microsoft.SharePoint.Client.File newFile = myListThumb.RootFolder.Files.Add(infoThumb);
      SPClientContext.Load(newFile);
      SPClientContext.ExecuteQuery();
      log.Info($”---- Image Thumbnail uploaded”);

      // Insert the metadata values back in the List
      myItem[“Title”] = myResult.description.captions[0].text;
      if (myResult.categories[0].detail.celebrities != null)
      {
        if (myResult.categories[0].detail.celebrities.Length > 0)
        {
          myItem[“PictureType”] = “Celebrity”;
            myItem[“PictureName”] = myResult.categories[0].detail.celebrities[0].name;
          myItem[“Gender”] = myResult.faces[0].gender;
          myItem[“Age”] = myResult.faces[0].age;
          myItem[“AdultContent”] = myResult.adult.isAdultContent.ToString();
          myItem[“Thumbnail”] = newFile.ServerRelativeUrl;

          myItem.Update();
          SPClientContext.ExecuteQuery();
            log.Info($”---- Image Analyze added to SharePoint Item”);
        }
        else
        {
          log.Info($”---- The image is not a celebrity”);
        }
      }
    }
  }
}
```

“se examina cada uno de los cambios y se obtiene un objeto con la imagen agregada”

Primero se crean dos objetos que contienen la Lista a utilizar en SharePoint y la Lista que contendrá los thumbnails. Luego se crea una consulta de cambio (variable “myChangeQuery”) que especifica que se requieren los cambios ocurridos en el ultimo minuto, que ocurren en elementos de la Lista y que sean del tipo “Add”, es decir, elementos nuevos:

```
public static ChangeQuery GetChangeQueryNew(string ListId)
{
  ChangeToken lastChangeToken = new ChangeToken();
   lastChangeToken.StringValue = string.Format(“1;3;{0};{1};-1”, ListId, DateTime.Now.AddMinutes(-1).ToUniversalTime().Ticks.ToString());
  ChangeToken newChangeToken = new ChangeToken();
  newChangeToken.StringValue = string.Format(“1;3;{0};{1};-1”, ListId, DateTime.Now.ToUniversalTime().Ticks.ToString());
    ChangeQuery myChangeQuery = new ChangeQuery(false, false);
  myChangeQuery.Item = true;  // Get only Item changes
  myChangeQuery.Add = true;  // Get only the new Items
  myChangeQuery.ChangeTokenStart = lastChangeToken;
  myChangeQuery.ChangeTokenEnd = newChangeToken;

  return myChangeQuery;
}
```

Luego de ejecutar la consulta, se examina cada uno de los cambios y se obtiene un objeto con la imagen agregada, que se convierte en un array de bytes por medio de la rutina “ConvertStreamToByteArray”:

```
public static Byte[] ConvertStreamToByteArray(ClientResult<System.IO.Stream> myFileStream)
{
  Byte[] bytReturn = null;

    using (System.IO.MemoryStream myFileMemoryStream = new System.IO.MemoryStream())
  {
    if (myFileStream != null)
    {
      myFileStream.Value.CopyTo(myFileMemoryStream);
      bytReturn = myFileMemoryStream.ToArray();
    }
  }

  return bytReturn;
}
```

En la misma rutina se llama a "GetAzureImageAnalyze", la que se encarga de hacer la consulta en Azure, utilizando como parámetro de entrada el array de bytes de la imagen. Esta rutina entrega de regreso un objeto del tipo "ImageAnalyzeResult" que contiene los resultados de la consulta y que tiene la forma:

```
#region ImageAnalyzeResult
public class ImageAnalyzeResult
{
    public Category[] categories { get; set; }
    public Adult adult { get; set; }
    public Tag[] tags { get; set; }
    public Description description { get; set; }
    public string requestId { get; set; }
    public Metadata metadata { get; set; }
    public Face[] faces { get; set; }
    public Color color { get; set; }
    public Imagetype imageType { get; set; }
}

public class Adult
{
    public bool isAdultContent { get; set; }
    public bool isRacyContent { get; set; }
    public float adultScore { get; set; }
    public float racyScore { get; set; }
}

public class Description
{
    public string[] tags { get; set; }
    public Caption[] captions { get; set; }
}

public class Caption
{
    public string text { get; set; }
    public float confidence { get; set; }
}

public class Metadata
{
    public int width { get; set; }
    public int height { get; set; }
    public string format { get; set; }
}

public class Color
{
    public string dominantColorForeground { get; set; }
    public string dominantColorBackground { get; set; }
    public string[] dominantColors { get; set; }
    public string accentColor { get; set; }
    public bool isBWImg { get; set; }
}

public class Imagetype
{
    public int clipArtType { get; set; }
    public int lineDrawingType { get; set; }
}

public class Category
{
    public string name { get; set; }
    public float score { get; set; }
    public Detail detail { get; set; }
}

public class Detail
{
    public Celebrity[] celebrities { get; set; }
    public Landmark[] landmarks { get; set; }
}

public class Celebrity
{
    public string name { get; set; }
    public Facerectangle faceRectangle { get; set; }
    public float confidence { get; set; }
}

public class Facerectangle
{
    public int left { get; set; }
    public int top { get; set; }
    public int width { get; set; }
    public int height { get; set; }
}

public class Landmark
{
    public string name { get; set; }
    public float confidence { get; set; }
}

public class Tag
{
    public string name { get; set; }
    public float confidence { get; set; }
}

public class Face
{
    public int age { get; set; }
    public string gender { get; set; }
    public Facerectangle1 faceRectangle { get; set; }
}

public class Facerectangle1
{
    public int left { get; set; }
    public int top { get; set; }
    public int width { get; set; }
    public int height { get; set; }
}
```

Luego, la rutina "GetAzureImageThumbnail" genera un thumbnail y lo retorna como un stream:

```
public static async Task<Stream> GetAzureImageThumbnail(byte[] myFileBytes)
{
    Stream streamReturn;
    Stream myFileStream = new MemoryStream(myFileBytes);

    // Create the API Client
     ComputerVisionClient computerVision = new ComputerVisionClient(
        new ApiKeyServiceClientCredentials(ConfigurationManager.AppSettings["azVisionApiServiceKey"]),
        new System.Net.Http.DelegatingHandler[] { });

    computerVision.Endpoint = "https://westeurope.api.cognitive.microsoft.com";

    // Get the thumbnail
     streamReturn = await computerVision.GenerateThumbnailInStreamAsync(
        40, 40, myFileStream, true);  // Size fixed to 40x40

    return streamReturn;
}
```

"permite enriquecer la información que los usuarios guardan en SharePoint encontrando metadatos sobre las imágenes"

Para guardar el thumbnail en la Biblioteca es necesario convertir el stream de regreso a un byte array utilizando un overwrite de "ConvertStreamToByteArray":

```
public static Byte[] ConvertStreamToByteArray(Stream myFileStream)
{
  Byte[] bytReturn = null;

  using (MemoryStream myFileMemoryStream = new MemoryStream())
  {
    if (myFileStream != null)
    {
      myFileStream.CopyTo(myFileMemoryStream);
      bytReturn = myFileMemoryStream.ToArray();
    }
  }

  return bytReturn;
}
```

6.– La rutina "GetAzureImageAnalyze" recibe como parámetros de entrada el array de bytes de la imagen y retorna un objeto con los valores encontrados por Azure:

```
public static async Task<ImageAnalyzeResult> GetAzureImageAnalyze(byte[] myFileBytes)
{
  ImageAnalyzeResult resultReturn = new ImageAnalyzeResult();

  HttpClient client = new HttpClient();

  // Request headers.
   client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", ConfigurationManager.AppSettings["azVisionApiServiceKey"]);

  // Request parameters
   string requestParameters = "visualFeatures=Categories,Description,Adult,Tags,Faces&language=en";

  // Assemble the URI for the REST API Call.
   string uri = ConfigurationManager.AppSettings["azVisionApiAnalyzeEndpoint"] + "?" + requestParameters;
  string contentString = string.Empty;

  HttpResponseMessage response;

  using (ByteArrayContent content = new ByteArrayContent(myFileBytes))
  {
    content.Headers.ContentType = new MediaTypeHeaderValue("application/octet-stream");

    // Execute the REST API call.
    response = await client.PostAsync(uri, content);

    // Get the JSON response.
      contentString = await response.Content.ReadAsStringAsync();
  }

  resultReturn = JsonConvert.DeserializeObject<ImageAnalyzeResult>(contentString);
  return resultReturn;
}
```

Cada consulta se realiza por medio de una llamada REST a un URL preespecificado del servicio de búsqueda (dado en el valor de la App Settings "azVisionApiAnalyzeEndpoint" y que es "https://westeurope.api.cognitive.microsoft.com/vision/v2.0/analyze"), utilizando como parámetros en el QueryString las propiedades que se desean recobrar (Categories, Description, Adult, Tags, Faces). En la App Settings "azVisionApiServiceKey" se mantiene el valor de la llave mencionada en el punto 4.

Por otro lado, la rutina "GetAzureImageThumbnail" produce el thumbnail:

```
public static async Task<Stream> GetAzureImageThumbnail(byte[] myFileBytes)
{
  Stream streamReturn;
    Stream myFileStream = new MemoryStream(myFileBytes);

  // Create the API Client
   ComputerVisionClient computerVision = new ComputerVisionClient(
     new ApiKeyServiceClientCredentials(ConfigurationManager.AppSettings["azVisionApiServiceKey"]),
    new System.Net.Http.DelegatingHandler[] {});

    computerVision.Endpoint = "https://westeurope.api.cognitive.microsoft.com";

  streamReturn = await computerVision.GenerateThumbnailInStreamAsync(
    40, 40, myFileStream, true);  // Size fixed to 40x40

  return streamReturn;
}
```

En este caso se esta utilizando el SDK, con el que se crea un cliente del tipo "ComputerVisionClient", al que se le entrega la llave del servicio y el endpoint del servicio. El método "GenerateThumbnailStreamAsync" genera el thumbnail, en este caso utilizando un tamaño fijo de 40x40 pixeles.

7.– Otras tres clases definen objetos utilizados por el WebHook:

```
public class ResponseModel<T>
{
  [JsonProperty(PropertyName = "value")]
  public List<T> Value { get; set; }
}

public class NotificationModel
{
  [JsonProperty(PropertyName = "subscriptionId")]
  public string SubscriptionId { get; set; }

  [JsonProperty(PropertyName = "clientState")]
  public string ClientState { get; set; }

  [JsonProperty(PropertyName = "expirationDateTime")]
  public DateTime ExpirationDateTime { get; set; }

  [JsonProperty(PropertyName = "resource")]
  public string Resource { get; set; }

  [JsonProperty(PropertyName = "tenantId")]
  public string TenantId { get; set; }

  [JsonProperty(PropertyName = "siteUrl")]
  public string SiteUrl { get; set; }
```

```
    [JsonProperty(PropertyName = "webId")]
    public string WebId { get; set; }
}

public class SubscriptionModel
{
    [JsonProperty(NullValueHandling = NullValueHandling.
Ignore)]
    public string Id { get; set; }

     [JsonProperty(PropertyName = "clientState", NullVa-
lueHandling = NullValueHandling.Ignore)]
    public string ClientState { get; set; }

    [JsonProperty(PropertyName = "expirationDateTime")]
    public DateTime ExpirationDateTime { get; set; }

    [JsonProperty(PropertyName = "notificationUrl")]
    public string NotificationUrl { get; set; }

     [JsonProperty(PropertyName = "resource", NullValue-
Handling = NullValueHandling.Ignore)]
    public string Resource { get; set; }
}
```

8.– Registre el WebHook en la Biblioteca de SharePoint y suba una imagen de un personaje mundial. El WebHook hará que la Función realice su trabajo, entregue los resultados encontrados por Azure, genere un thumbnail y lo suba a la Biblioteca indicada y, finalmente, muestre los metadatos de la imagen:

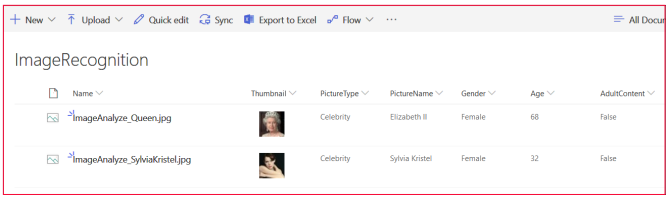

Imagen 2.- Imágenes con thumbnail y metadatos.

Conclusiones

El servicio de Reconocimiento de Imágenes del Azure Computer Vision API permite enriquecer la información que los usuarios guardan en SharePoint encontrando metadatos sobre las imágenes y creando thumbnails. El servicio de Reconocimiento de Imágenes es fácil de utilizar desde cualquiera lenguaje de programación, y produce resultados confiables rápida y seguramente. El API utiliza algoritmos de Inteligencia Artificial que se mejoran con el uso, por lo que no es necesario crear ni entrenar algoritmos propios.

GUSTAVO VELEZ
MVP Office Apps and Services
gustavo@gavd.net
http://www.gavd.net

22

Overview a Azure Batch AI

La inteligencia artificial ya hace un tiempo que están entre nosotros, ¿quién no ha oído hablar de los cognitive services, bots, machine learning o ML.Net entre otros? La base de la AI son los modelos de datos que la alimentan, estos modelos suelen ser diseñados por expertos en AI o Data Scientists. Pero estos modelos hay que entrenarlos y para hacerlo se requiere de mucho hardware y tiempo.

Azure Batch AI es un servicio PaaS de Azure creado para ayudar a entrenar y probar estos modelos en Azure sin la necesidad de administrar la complejidad de la infraestructura que requiere este tipo de procesos.

Con Azure Batch AI de lo único que nos debemos preocupar es de:

- Seleccionar los recursos necesarios.
- Donde almacenar los inputs de entrada del modelo.
- Donde almacenar los outputs de salida del modelo.

Usos de Azure Batch AI

Se puede usar Azure Batch AI en dos escenarios:

1.– Crear una instancia para entrenar, probar y puntuar modelos de AI y aprendizaje automático en clusters utilizando GPUs o CPUs.

2.– Seleccionando un cluster como destino de un workflow con Azure Machine Learning u otro servicio o herramienta de Azure AI.

Training

Como hemos comentado, Azure Batch AI nos ayuda a entrenar modelos de Machine Learning como redes neuronales y/u otras aproximaciones del mundo AI. Además, soporta las librerías open source más conocidas y extendidas en el mundo AI como TensorFlow, PyTorck, Chainer o Microsoft Cognitive Toolkit (CNTK).

"La base de la AI son los modelos de datos que la alimentan, estos modelos suelen ser diseñados"

Azure Batch AI nos ayuda a entrenar nuestros modelos gracias las siguientes características:

- Distribute Training: Permite escalar verticalmente jobs en diferentes GPUs dentro de una misma red permitiendo entrenar grandes redes con grandes volúmenes de datos.
- Experimental: Permite entrenar varios trabajos a la vez, escalado horizontal, permitiendo probar nuevas ideas o ajustar y/u optimizar los Jobs.
- Execute in Parallel: Nos permite puntuar o entrenar varios modelos en paralelo permitiendo que los trabajos se realicen más rápidamente.

Cómo funciona

Azure Batch AI se puede crear y administrar a partir de:

- SDK de Azure Batch AI.
- Azure CLI.
- Powershell.
- Azure Portal

Estas herramientas no permitirán:

- Aprovisionar y escalar clusters: Podemos elegir el tipo de VM que queremos utilizar, así como el número de nodos, también podemos configurar escalado vertical de forma automática o manual.
- Administración de las dependencias y los contenedores: Por defecto las VM que utiliza el servicio son VM Linux con una configuración predeterminada. Esta configuración puede modificarse o extenderse mediante scripts de inicio y se pueden personalizar imágenes en caso de querer utilizar otro tipo de VM.
- Distribución de datos: Para administrar los datos de entrada, salida y scripts, se puede utilizar:
  - Azure Files
  - Azure Blob Storage
  - Servidor NFS

  También permite soluciones de almacenamiento personalizadas que pueden configurarse en los nodos del cluster y los contenedores.
- Programación de trabajos: Se pueden encolar trabajo y priorizarlos, de esta forma se pueden compartir recursos.
- Control de errores: Podemos ver en qué estado están nuestros Jobs y en caso de error podemos reiniciarlos.

"Azure Batch AI nos ayuda a entrenar modelos de Machine Learning como redes neuronales"

Jerarquía de recursos

Al crear el servicio de Azure Batch AI en Azure, este se creará en uno o más recursos. Cada recurso creado contendrá una estructura como la que muestra la siguiente imagen:

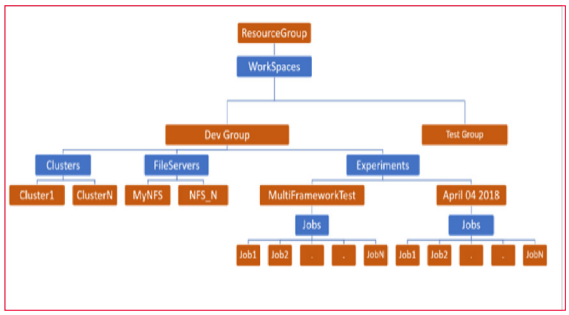

Imagen 1.- Jerarquía de recursos.

- WorkSpace: Los WorkSpaces ayudan a separar el trabajo en diferentes grupos o proyectos. Es una colección de alto nivel del resto de recursos.
- Clúster: Es la agrupación de VM que ejecutarán los Jobs. El tamaño de las VM en un cluster es el mismo, así como todas son del mismo tipo. Generalmente se configuran diferentes clusters para cada categoría o tipo de trabajo a realizar.
- Servidor de archivos: Se puede crear un servidor de archivos donde almacenar datos de entrada, de salido o los scripts de aprendizaje. Por defecto se crea un servidor de ficheros NFS administrado en un solo nodo que se pueden montar en los diferentes clusters proporcionando una ubicación de almacenamiento centralizada, accesible y sencilla. Si esta opción no cumple con las necesidades también se pueden configurar otras opciones como Azure Storage, Azure Files o alguna solución personalizada.
- Experimento: Agrupación de Jobs relacionados que se administran conjuntamente y que intentan resolver un problema específico.
- Job: Son las tareas o scripts que entrenarán nuestros modelos. Cada job ejecuta un único script en un cluster y workspace

Creando nuestro primer servicio Azure Batch AI

Una vez ya tenemos la visión global de lo que el servicio nos proporciona y como se estructura vamos a crear nuestro primer servicio. Como ya hemos comentado se puede crear mediante:

- Azure CLI.
- Powershell.
- Portal.

En nuestro caso vamos a ver como se crea mediante Azure CLI del Portal:

- Entramos en el portal y abrimos la CLI:

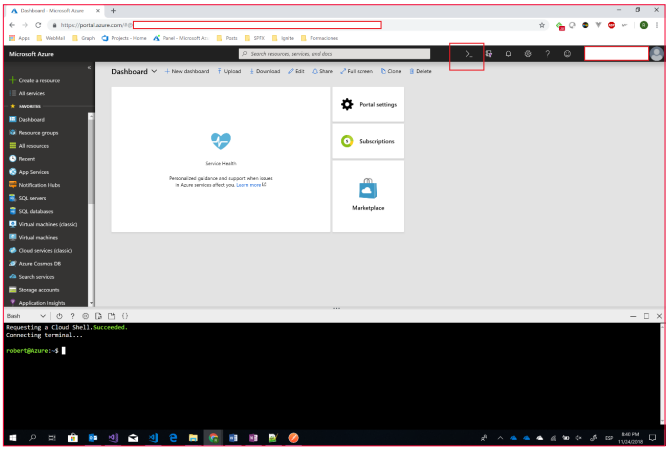

Imagen 2.- Acceso al entorno de Azure CLI.

- Ahora creamos el recurso que vamos a usar:

```
az group create \
  --name compartimoss \
  --location westeurope
```

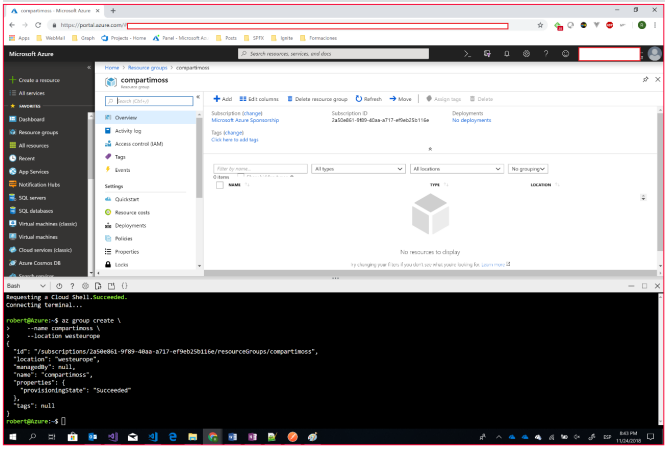

Imagen 3.- Creación de los recursos necesarios

- Ahora crearemos el servicio de Azure Batch AI dentro de este recurso:

```
az batchai workspace create \
  --workspace compartimossai \
  --resource-group compartimos
```

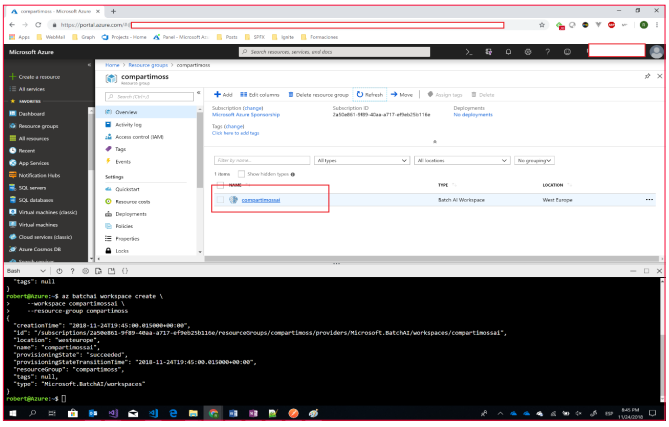

Imagen 4.- Creación del servicio de Azure Batch AI en el grupo de recursos.

- Una vez tenemos creado el workspace creamos el cluster con las VM que utilizaremos para el entrenamiento, en nuestro caso VM Stantdard_NC6:

```
az batchai cluster create \
  --name compartimossaicluster \
  --workspace compartimossai \
  --resource-group compartimoss \
  --vm-size Standard_NC6 \
  --target 1 \
  --user-name rbermejo \
  --generate-ssh-keys
```

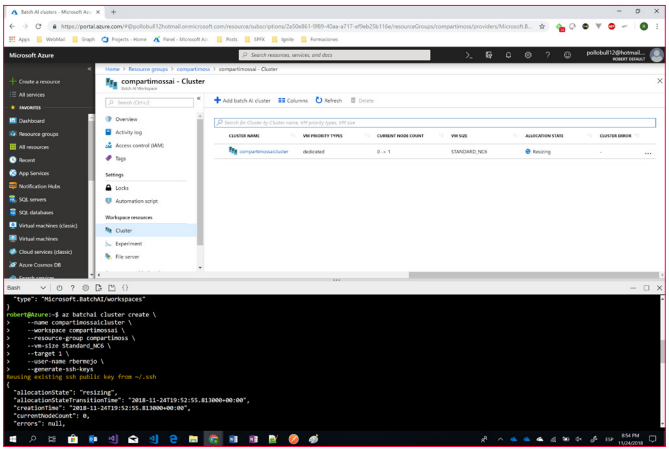

Imagen 5.- Creación del clúster de con las VM necesarias.

Como se ve en la imagen cuando se crea el cluster este se crea en estado resazing, al cluster estará creado y activo cuando pase a estado Steady. Ahora ya tenemos el entorno creado para poder empezar a trabajar con él.

Conclusiones

En este artículo se ha intentado hacer un overview de que es Azure Batch AI, cuáles son sus características, que nos proporciona y como lo podemos crear.

Podemos ver que es un servicio muy potente que nos ayudará a la hora de conseguir recursos para entrenar nuestros modelos de forma rápida y controlando nuestros costes, lo que nos permitirá reducir de forma exponencial el tiempo de mejora de estos.

ROBERT BERMEJO
Cloud Architect & Technical Lead Consultant in TOKIOTA
Microsoft Azure MVP
bermejoblasco@live.com
@robertbemejo
www.robertbermejo.com

25

Data Lake Analitycs y U-SQL

El reducto del programador C# en Big Data

No aun no me he vuelto loco (darme tiempo), y no, no quiero aportar al mundo un artículo de protesta o queja alguna, pero si me gustaría dejar constancia con las siguientes páginas, de que vivimos en un mundo tecnológico apasionante, lleno de evolución y de cambios, pero que a su vez no deja enemigos vivos. Un claro ejemplo es toda la rama Data Plattform, AI y Big Data que ya está en nuestro día a día, y que sin darnos cuenta nos rodea, nos absorbe y nos hace evolucionar.

En el pasado SQL Saturday de Madrid, pude ver como lenguajes T-SQL, C# o DAX, van perdiendo peso respecto a otros como SCALA o Python, y no hace más que marcar la tendencia de Microsoft a abrir su abanico y sus tools de desarrollo incorporando tecnologías como Databricks o Hadoop a su oferta. Pero aún nos queda un pequeño reducto a los programadores de C#, que incluso también puede resultar cómodo a los programadores de T-SQL, y no es otro que Data Lake Analitycs y el lenguaje U-SQL.

¿Qué es Azure Data Lake Analitycs?

Data Lake Analitycs, lo encontramos como un servicio de Azure en modo PaaS, y que se engloba dentro de Data Lake. Azure Data Lake lo podríamos entender como un catálogo de servicios y herramientas que Azure nos ofrece y que da servicio tanto a desarrolladores como a científicos de datos, permitiendo almacenar datos de cualquier tamaño, forma y tipo, sin importar su volumen.

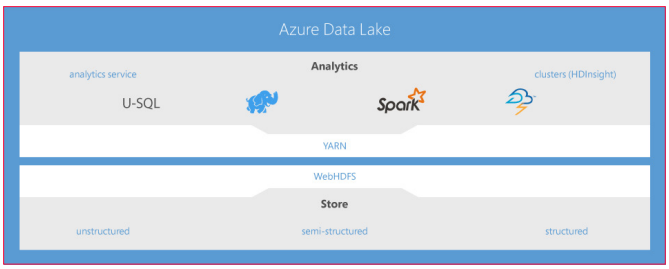

Imagen 1.- Data lake.

Dentro de los servicios como hemos dichos encontramos Spark, Hadoop y por supuesto el que nos ocupa Data Lake Analytics Service. Se puede decir que Data Lake Analytics nos va a permitir analizar mediante la ejecución de trabajos en U-SQL, aquella información ya sea estructurada como una tabla en SQL, o semiestructurada como un fichero de texto, imágenes, y que almacenaremos en Data Lake Store.

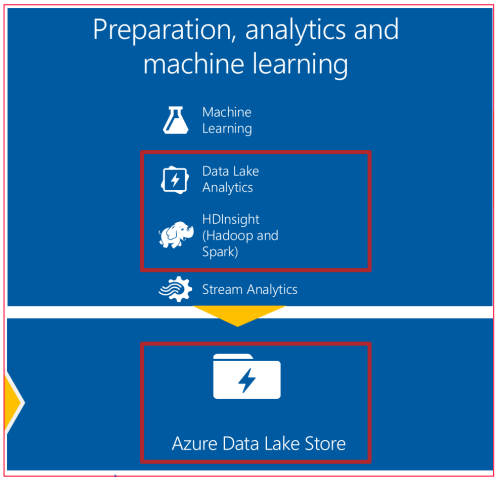

Imagen 2.- Data Lake Store.

U-SQL, conociendo el lenguaje

Todos los ejemplos que vamos a ir viendo a continuación parten de unos datos almacenados en un Data Lake Store, y mediante el diseño de trabajos con U-SQL, podremos analizar dichos datos. Por ello, aunque no lo expliquemos en este artículo, recordar que es necesario tener creado el servicio de Data Lake Store, para poder empezar a trabajar con Data Lake Analitycs. Aunque es cierto que podríamos usar otros orígenes de datos como un Blob Storage o un Azure Data Warehouse.

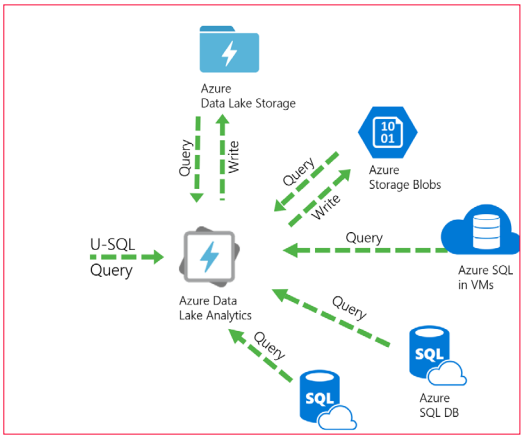

Imagen 3.- Múltiples orígenes de datos.

Lo más importante es empezar a familiarizarnos con U-SQL,

aunque ya anticipo que es muy intuitivo y en pocos minutos estamos trabajando con soltura. En origen U-SQL nació en Microsoft, y se pensó en él como un lenguaje sobre el que ejecutar el Framework de Big Data.

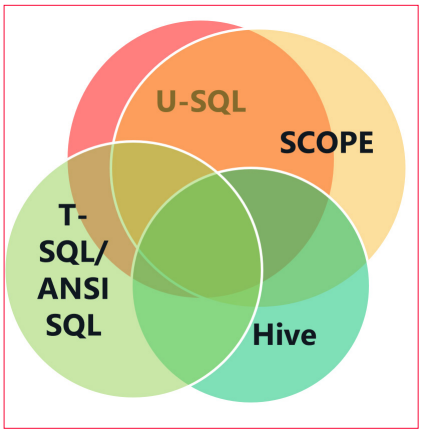

Imagen 4.- U- SQL Scope.

Incluía todo lo bueno del T-SQL, acercando así a los desarrolladores de SQL Server, y por otro lado a los programadores de C#, permitiendo dar forma a un lenguaje poderoso con muchas similitudes en sintaxis tanto con SQL como con lenguajes de desarrollo en .Net. Tranquilos que no es una mezcla de T-SQL con C#, sin orden, sino que permite declarar un .USQL, que en sintaxis es muy similar a SQL, e incorporar DLL 's en C# y hacer referencias a funciones y objetos.

Desarrollando el primer código .USQL

Vamos a explicar un pequeño código en U-SQL, que obtiene desde un fichero CSV alojado en Data Lake Store, todos los premiados, y ganadores de los premios Oscars, desde su creación.

```
@data =
EXTRACT year      string,
category    string,
winner      string,
entity      string

FROM @"/Oscars/dataOscars.csv"
USING Extractors.Csv(silent:true);
```

Como podemos observar en el código anterior, definimos un proceso de extracción con la sentencia EXTRACT, e indicamos los campos de dentro del CSV que queremos mapear en una variable @data. En este caso estamos usando un Extractor ya existente como es el de CSV, pero más adelante veremos cómo podemos crearnos nuestros propios extractores.

El siguiente fragmento del script, nos sirve para trabajar en memoria con el resultado de la extracción, que recordemos estaba en @data. En este caso se parece mucho a una SELECT en T- SQL, y simplemente vamos a filtrar los datos por el año 2000, asignándosela a la variable @result

```
@result =
SELECT year,category,winner,entity
FROM @data
WHERE year == "2000";
```

Por último, tenemos que dar una salida a nuestra query, y para eso usamos la sentencia OUTPUT, y un operador de salida como Outputters.csv, que en este caso nos parsea el dato obtenido a un fichero CSV en nuestro Store.

```
OUTPUT @result
TO @"/Oscars/resultFiltered.csv"
USING Outputters.Csv();
```

Como aclaración, hay que recordar que el dato que recibe nuestro operador de salida, en este caso al ser un CSV, tiene que ser una colección de datos, o lo que es lo mismo debe tener estructura de tabla.

Ejecutando trabajos desde el portal de Azure

Una vez hemos visto un ejemplo muy sencillo de proceso en U-SQL, vamos a ver como ejecutar este en Data Lake Analytics. Para ello debemos acceder al servicio en el portal de Azure, y seleccionar el apartado New Job. Como podemos ver en la imagen siguiente, basta con pegar el código anterior en el editor de trabajos, darle un nombre a nuestro nuevo JOB y elegir las unidades de procesamiento AUs.

Antes de poner un número de AUs a lo loco debemos entender que esto nos va a permitir paralelizar nuestros procesos, en tanto que añadamos más unidades de procesamiento.

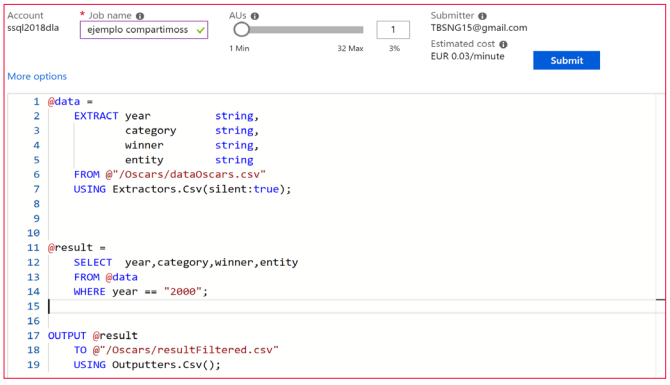

Imagen 5.- Portal Data Lake.

" no es una mezcla de T-SQL con C#, sin orden, sino que permite declarar un .USQL, que en sintaxis es muy similar a SQL, e incorporar DLL 's en C# y hacer referencias a funciones y objetos"

Como vemos en la siguiente imagen, un proceso de Extracción por ejemplo de una biblioteca de imágenes puede tener implícito 70 extracciones por ejemplo porque tenga varios ficheros. Cada proceso lo podemos representar como un vértice, por lo que en este caso si sería bueno procesar en paralelo ya que reduciríamos el tiempo global de la extracción.

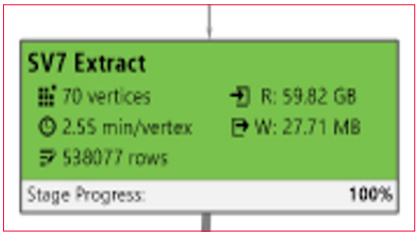

Imagen 6.- AU, Vertex.

Dicho todo esto, tener en cuenta que a mayor número de procesos en paralelo, mayor coste tiene la ejecución de este trabajo, ya que el principal coste de este servicio es en ejecución. Para nuestro ejemplo nos vale con configurar un único AuS, y el coste por minuto sería de 0,03 € / min

Para comprobar si todo es correcto, y no tenemos errores en el código, pulsaríamos sobre Submit y este empezaría su ejecución.

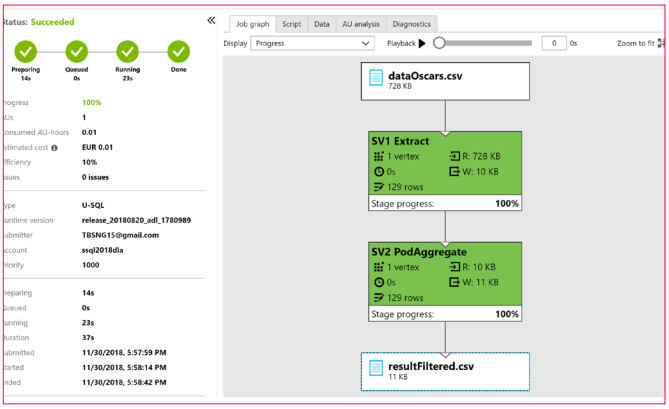

Imagen 7.- Ejecución de un trabajo.

Si todo es correcto, podremos ver un mapa del proceso y el tiempo de ejecución de cada fase. Además, desde el resumen de ejecución también podemos ver el fichero de salida, en este caso un CSV que filtra por el año 2000 todos los premios Oscars.

0	1	2	3
2000	ACTOR IN A LEADING ROLE	False	Javier Bardem
2000	ACTOR IN A LEADING ROLE	True	Russell Crowe
2000	ACTOR IN A LEADING ROLE	False	Tom Hanks
2000	ACTOR IN A LEADING ROLE	False	Ed Harris
2000	ACTOR IN A LEADING ROLE	False	Geoffrey Rush
2000	ACTOR IN A SUPPORTING ROLE	False	Jeff Bridges
2000	ACTOR IN A SUPPORTING ROLE	False	Willem Dafoe
2000	ACTOR IN A SUPPORTING ROLE	True	Benicio Del Toro
2000	ACTOR IN A SUPPORTING ROLE	False	Albert Finney

Imagen 8.- Resultado con premios filtrados por año.

Extendiendo con código C# nuestros trabajos en Visual Studio: UDOs

Bueno por ahora hemos visto como ejecutar trabajos de análisis de datos desde el servicio contra nuestro almacén de datos. Pero ¿dónde está el prometido código en C#? ¿Qué nos puede aportar? No siempre vamos a poder alcanzar nuestro objetivo con las operaciones que U-SQL nos trae de base, y podemos llegar a extender esta funcionalidad con código personalizado en este caso en C#.

Una primera forma para poder incorporar código C# a nuestro U-SQL es mediante los User – define – operations (UDOs).

Estos nos permiten extender las operaciones básicas como extracciones, operadores de salida, procesadores…, algunos ya los hemos visto como el extractor de CSV o el operador de salida hacia CSV. Antes de nada, deberemos instalar las tools de Azure Data Lake y Stream Analytics, para poder depurar y ejecutar desde nuestra máquina local.

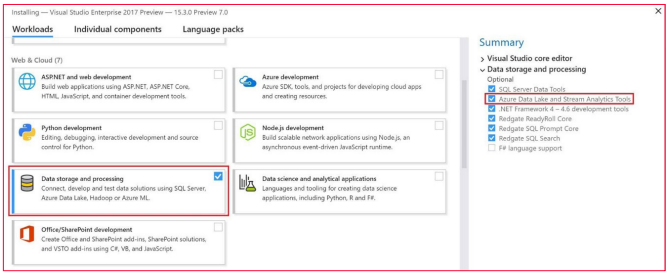

Imagen 9.- Data storage and processsing.

Para empezar, vamos a escribir el código U-SQL, que se va poder dividir en tres fases:

1.– Extracción: Mismo código que en el ejemplo anterior, extraemos los datos desde el CSV de los premios Oscars.

```
@data =
  EXTRACT year      string,
      category    string,
      winner      string,
      entity      string
  FROM @"Oscars/dataOscars.csv"
  USING Extractors.Csv(silent:true);
```

2.– Procesamiento: Este operador no lo utilizamos antes, y ahora nos va a permitir invocar un proceso en backend C#, y modificar nuestra colección de datos @data.

```
@dataSentences =
  PROCESS @data
  PRODUCE year string,
      category string,
      winner string,
      entity string,
      sentence string
  USING new USQL_UDO.ActorDetails();
```

Como novedad debemos fijarnos en la instrucción PROCESS y PRODUCE, que nos van a permitir procesar la colección @data con el proceso que inyectamos

con la sentencia USING, y que va a ser nuestro UDO de procesamiento.

3.– Filtrado y procesado: Esta fase de ejecución es muy similar al primer ejemplo, filtramos por el campo sentence para asegurarnos que no es nulo, y lo exportamos a un fichero CSV de salida en el Store.

```
@dataSentencesWithData =

  SELECT year,
      category,
      winner,
      entity,
      sentence

  FROM @dataSentences
  WHERE sentence != "";

OUTPUT @dataSentences
   TO "Oscars/resultSentences.csv"
   USING Outputters.Csv();
```

Construyendo el procesador U-SQL UDO Actor Details

Una vez como hemos visto como instanciar un UDO en un código U-SQL, vamos a ver brevemente como crearlo en Visual Studio. Lo primero es crear un proyecto del tipo U-SQL Project, para poder dar de alta nuestro código de extensión.

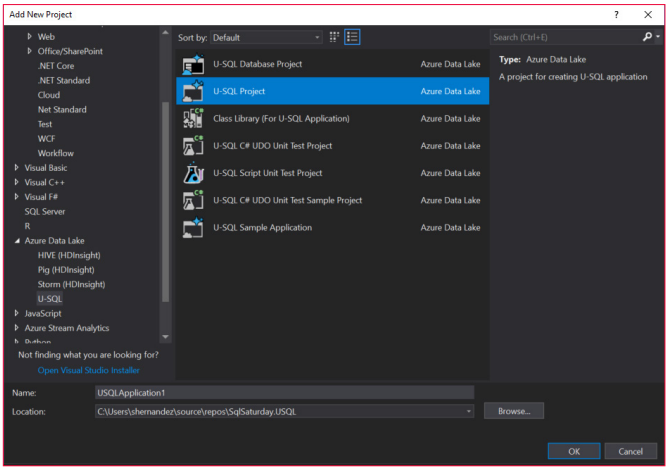

Imagen 10.- Proyecto en Visual Studio.

Añadimos un nuevo elemento del tipo U-SQL Script, y sobre este en el Solution Explorer con el botón derecho seleccionamos Add-> User Code -> Add C# UDO . Esta acción no va a generar un .cs relacionado con el .usql, este nuevo .cs va a tener el código en C# que podremos añadir a nuestro script U-SQL.

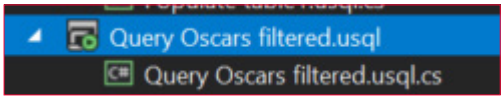

Imagen 11.- UDO .cs

En este caso como nuestro UDO va a extender a un PROCESSOR, debemos tener una clase en C# que implemente la interfaz IProcessor.

```
namespace USQL_UDOagen
{
public class ProcessosTesting : IProcessor
```

Nuestra clase básicamente va a tener un método Process,- que va a recorrer todas las filas de nuestro CSV en memoria, y si la película se encuentra en un diccionario de frases célebres, va a añadir una columna SENTENCE a nuestra colección, con la frase en cuestión.

```
public override IRow Process(IRow input, IUpdatableRow output)
    {
      string year = input.Get<string>("year");
      string category = input.Get<string>("category");
      string winner = input.Get<string>("winner");
      string entity = input.Get<string>("entity");
      string sentence = string.Empty;

      var current = string.Empty;
      if (Sentence.TryGetValue(entity, out current))
      {
        sentence = current;
      }

      output.Set<string>(0, year);
      output.Set<string>(1, category);
      output.Set<string>(2, winner);
      output.Set<string>(3, entity);
      output.Set<string>(4, sentence);
      return output.AsReadOnly();

    }
```

Este método es invocado por nuestro script U-SQL una vez por cada fila en la colección de datos, así conseguimos iterar. Si volvemos al script U-SQL podemos recordar que la forma de instanciar este UDO, es invocarlo como si incluyéramos un objeto de clase en C#, con la sentencia USING.

```
USING new USQL_UDO.ActorDetails();
```

Por último, solo nos queda probarlo y depurarlo

Podemos ejecutarlo en local, o contra el servicio de Azure Data Lake, todo desde el Visual Studio. Pero primero vamos a depurarlo, para ello necesitamos que tengamos seleccionado en el Solution Explorer el U-SQL Script a ejecutar, se marcará con un símbolo de Play, y posteriormente seleccionar F5 o Debug en la ribbon superior.

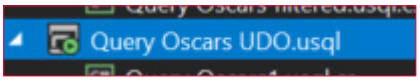

Imagen 12.- Depuración del script.

Se va a inicializar una aplicación de consola si todo es correcto, y si añadimos un brakepoint en el .cs de nuestro UDO recién implementado, veremos que se detiene la ejecución, tal y como estamos acostumbrados con aplicaciones de backend en .Net

```
public override IRow Process(IRow input, IUpdatableRow output)
{
    string year = input.Get<string>("year");
    string category = input.Get<string>("category");
    string winner = input.Get<string>("winner");
    string entity = input.Get<string>("entity");
    string sentence = string.Empty;

    var current = string.Empty;
    if (Sentence.TryGetValue(entity, out current))
    {
        sentence = current;
    }
```

De este modo podemos depurar e inspeccionar nuestros scripts en tiempo real, sin necesidad de añadir trazas o depuración en remoto. Si el resultado es correcto, en el propio Visual Studio podremos ver los Outputs, en este caso un fichero .csv

> ***" tener en cuenta que a mayor número de procesos en paralelo, mayor coste tiene la ejecución de este trabajo, ya que el principal coste de este servicio es en ejecución"***

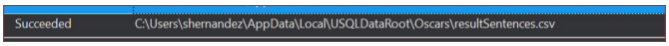

Imagen 13.- Output de VS.

Como nota muy importante para que funcione en local, y podamos simular el Data Lake Store, deberemos configurar un path del cual van a extraerse los ficheros, y se van a dejar los outputs. Para ello en Tools->Options->Azure Data Lake, deberemos configurar el fichero DataRoot, tal cual se ve en la siguiente figura.

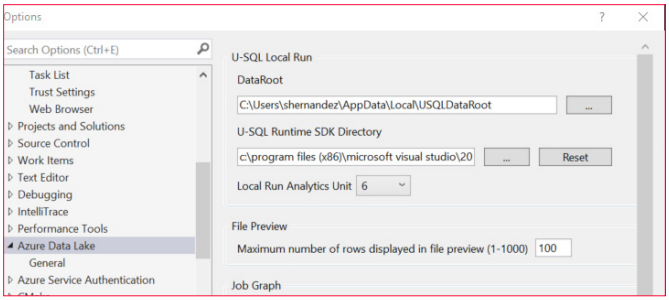

Imagen 14.- Configurando el entorno.

Ejecutando el trabajo desde el Visual Studio

Como hemos dicho en el apartado de depuración podemos ejecutar el Script bien contra el servicio de Azure, o contra el simulador de Data Lake en local. En cualquier caso, escogemos el script a ejecutar del solution explorer, y observamos que sobre el editor del fichero U-SQL se nos presenta la siguiente ribbon:

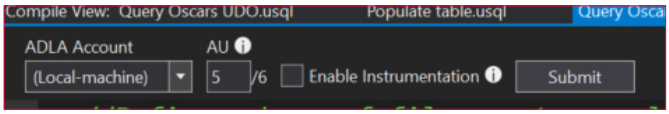

Ilustración 15.- Ribbon submit.

Casi ya no es ni necesario explicarla, ya que es muy parecido al panel de configuración de trabajos que vimos en Azure, simplemente añade el desplegable para escoger entre una cuenta Azure Data Lake o en Local. Hay que recordar que, si escogemos una cuenta de Azure, se nos va a cobrar por ejecución, ya que realmente estamos mandando la ejecución de un nuevo trabajo. Vamos a seleccionar local-machine, vemos que por defecto nos permite como máximo 6 unidades de procesamiento o AU, y seleccionamos Submit para arrancar la ejecución en local.

Si la compilación es correcta, se nos va a arrancar de nuevo la consola de ejecución, y se nos va a presentar además de los resultados de salida como en la depuración, el mapa de ejecución y el detalle de tiempos, a un estilo muy similar del primer ejemplo sobre Azure.

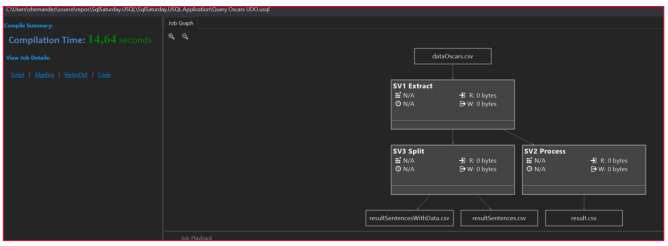

Imagen 16.- Resumen de ejecución en local.

Conclusiones de Azure Data Lake Analytics

Evidentemente con este artículo estamos muy lejos de dominar el servicio, ya que nos queda mucho que tratar, pero ya podemos ver un poquito mi idea inicial, que no era otra de dar una herramienta de trabajo y más que útil a los desarrolladores de C# que necesiten hacer análisis de datos. Debo de ser sincero, y no sé cómo va a evolucionar esto, ya que otras herramientas como Spark o Databricks están pasando como cohetes, y posiblemente Data Lake Analitycs no tenga mucho más recorrido, pero con lo que tenemos ya podemos plantearnos hacer trabajos interesantes.

Como empecé diciendo, es nuestro reducto, y si lo utilizamos posiblemente lo mantengamos más de lo que nos creemos, ¡¡¡¡y si no siempre tendremos el Python o Scala for Dummies!!

SERGIO HERNÁNDEZ MANCEBO
Principal Team Leader en ENCAMINA
Azure MVP
@shmacenbo

Mentoring

Comparti*MOSS*

Un servicio experto alrededor de su SharePoint

CompartiMOSS le puede ayudar a través de su programa de Mentoring!

Contacte con nosotros y le enviaremos los planes de mentoring que tenemos disponibles para SharePoint.

31

Entrevista ENCAMINA

encamina
PIENSA EN COLORES

Transformamos digitalmente a las medianas y grandes empresas con soluciones colaborativas, de productividad e inteligencia artificial, basadas en la mejor tecnología Microsoft.

Nos apasiona trabajar de forma fresca, optimista y en equipo. Nos gusta pensar en colores.

¿Cuál es el origen del nombre de la empresa?

Nacimos llamándonos Pharus Innovaciones, pero pronto constatamos que era un nombre poco usable y de corto recorrido. Así pues, a los 4 años de vida, abrimos un concurso interno, y con la ayuda de un estudio especializado en branding, decidimos que ENCAMINA comunicaba el espíritu de colaboración y futuro que no ponía ningún límite a nuestra misión. Por otro lado, nuestro refrán "Piensa en Colores" nació hace 10 años, porque queríamos contagiar una actitud fresca, optimista y comprometida que utiliza el ingenio y la creatividad para encontrar soluciones de tecnología y talento que mejoren el presente de las personas, la empresa y nuestra sociedad.

¿Por qué y cómo empezó en el mundo de la tecnología?

A los socios fundadores les apasionaba la tecnología y los proyectos retadores, pero estaban cansados de llevar años viviendo fuera de casa para poder disfrutar de ese mundo. Tanto es así, que un párrafo de la Visión actual de ENCAMINA dice de "…ser un líder a nivel nacional y un referente internacional, desde el lugar donde más nos gusta vivir…".

Los comienzos fueron muy distintos a lo que somos hoy (trabajábamos fundamentalmente en el entorno financiero), pero diría que SharePoint y su evolución nos ha estado acompañando desde su creación, hace ya más de 15 años.

¿Cuáles son las principales actividades tecnológicas hoy en día?

Fundamentalmente abordamos proyectos que aportan Innovación para el Negocio de nuestros clientes en los ámbitos de Modern Applications (C#/JS + Azure PaaS) e Inteligencia Artificial (ML y Cognitive Services, Bots), Data (Databricks, BigData, IoT) e Integración (LogicApp, Functions & APIs) y también proyectos en la que es nuestra especialidad histórica: Productividad (Collaboration & BI) y Cliente Digital (CRM & Multicanalidad).

Por otro lado, nuestros Managed Services se centran en la Operación, Optimización y Mejora de la Infraestructura Cloud (sobre Azure) y el mantenimiento y Gestión del Ciclo de vida de aplicaciones (en .NET).

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

Nuestro propósito como empresa es "Crecer personal y profesionalmente, inspirar y ayudar a otros, transformar a nuestros clientes y crear valor para el mundo."

En la práctica significa que nuestra RSC nos hace volcarnos al desarrollo del talento TIC en la sociedad, desde los muy jóvenes en las escuelas, pasando por nuestra propia gente, hasta las comunidades técnicas más potentes. Y en ese propósito volcamos mucho esfuerzo, en ocasiones ayudados por nuestro motor de marketing, donde ponemos mucho cariño también y con el que aspiramos a contagiar

positivamente a nuestro entorno con cierta dosis de frescura y optimismo. La parte del valor para el mundo, además de todo lo anterior y por los proyectos que realizamos para nuestros clientes, también aportamos por medio de nuestro laboratorio de innovación, que nosotros llamamos ENESFERA, donde no sólo se construyen productos digitales, sino que también se promueve la innovación como espíritu dentro y fuera de ENCAMINA.

¿Cuáles son las actividades que realiza en la comunidad técnica?

Somos asiduos patrocinadores de eventos técnicos y mucha de nuestra gente colabora con Comunidades organizando eventos, ayudando a coordinar, como speakers o simplemente asistiendo. Generemos multitud de contenidos técnicos para la comunidad en forma de Webinars, conferencias, blogs, artículos, libros blancos, Frameworks abiertos, etc. Y también nos gusta ayudar en los foros más jóvenes, acogiendo alumnos de primaria y la ESO, hasta intercambios de la Unión Europea, pasando por becas, cátedras, formaciones duales, ciclos, posgrado, etc.

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

La tecnología será exponencialmente más potente, y a la vez seguirá siendo cada vez más y más compleja, por lo que necesitaremos el mejor talento para dominarla. Creemos que hay grandes posibilidades y esperanza en el futuro gracias a la tecnología, pero la ética y los valores deben regir su uso y evolución. Creemos que la filosofía de Pensar en Colores tendrá mucho sentido también en el futuro.

Desde ENCAMINA, tenemos especial interés en el impacto de la Inteligencia Artificial desde el día a día de las aplicaciones y los procesos de negocio hoy a los agentes autónomos del futuro cercano.

33

Azure Cognitive Services en un contenedor

No todo en Inteligencia Artificial es trabajar con Machine Learning definiendo y entrenando nuestros propios algoritmos para resolver el problema que nos plantean desde negocio. Existen diversos servicios con inteligencia artificial pre-construida que nos pueden ayudar a acercarnos a esos escenarios que no necesitan de algún tipo algoritmo específico. Me estoy refiriendo a los servicios cognitivos de Microsoft y las diferentes posibilidades que nos ofrece para hacer nuestras aplicaciones más inteligentes.

Imagen 1. – Ámbitos de problemas resueltos por servicios cognitivos de Microsoft.

Estos servicios nos permiten consumir un modelo de inteligencia artificial que es capaz de analizar o predecir el contenido que le enviemos a través de su interfaz REST. Ahora bien, ¿qué pasa en aquellos escenarios que por privacidad no podemos hacer compartir esa información con un servicio externo? ¿qué pasa en escenarios con conectividad escasa de internet? ¿qué hacemos con los escenarios en los que tenemos que resolver al instante y no podemos esperar a que los datos suban a la nube y vuelva nuestra predicción?

> ***"Existen diversos servicios con inteligencia artificial pre-construida que nos pueden ayudar"***

Siguiendo con la misma filosofía de utilizar inteligencia artificial pre-construida, ahora podemos ejecutar estos servicios cognitivos en un contenedor, casi de forma aislada, para que podamos hacer predicciones o análisis de nuestros datos en el Edge o manteniendo los datos bajo nuestro control, cumpliendo con la legislación necesaria.

¿Qué servicios cognitivos tenemos disponibles como contenedores?

Desde el equipo de producto de Azure Cognitive Services, han dispuesto los siguientes contenedores:

- Text Analytics Containers, con los servicios de extracción de frases claves, detección de idioma y análisis de sentimiento.
- Face Containers que permite hacer una detección de caras, verificación y análisis de sentimiento.
- Recognize Text Containers, un servicio de visión, para detectar y extraer texto impreso de imágenes.

Probamos el Text Analytics Container

Para empezar con estos contenedores, necesitamos algún servicio local o remoto que permita ejecutar contenedores Docker, además, este debe permitir conectarse a Azure para que se pueda enviar los datos de facturación del servicio y validar su clave.

Dependiendo del servicio que vayamos a utilizar, tendremos unos requisitos mínimos para el contenedor, por ejemplo, para ejecutar el contenedor de Key Phrase Extraction necesitamos como mínimo 1 core y 2GB de memoria y como recomendado 1 core y 4GB de memoria. Tenedlo en cuenta porque, además, dependiendo del número de peticiones que hagamos, vamos a tener que ser capaces de escalar horizontalmente el número de instancias del contenedor.

Los contenedores se encuentran en un servicio de registro de Microsoft, para el Key Phrase Extraction tenemos disponible la imagen en la siguiente URL: mcr.microsoft.com/azure-cognitive-services/keyphrase.

Si vamos a trabajar desde local, tenemos que hacer un Docker pull para que se descargue la imagen y la tengamos disponible para instanciarla.

docker pull mcr.microsoft.com/azure-cognitive-services/keyphrase:latest

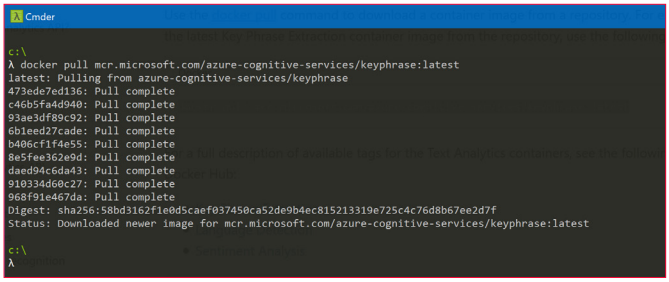

Imagen 2.– Descargando la imagen del contenedor Key Phrase.

> ***"consumir un modelo de inteligencia artificial que es capaz de analizar o predecir el contenido que le enviemos a través de su interfaz REST"***

Ahora toca instanciar el contenedor para tener accesible el servicio web que nos permite ejecutar el modelo de extracción de palabras claves. Para ello, ejecutaremos el siguiente comando de docker:

```
docker run --rm -it -p 5000:5000 --memory 8g --cpus 1 mcr.
microsoft.com/azure-cognitive-services/sentiment  Eula=ac-
cept  Billing=https://westus.api.cognitive.microsoft.com/text/
analytics/v2.0 ApiKey=0123456789
```

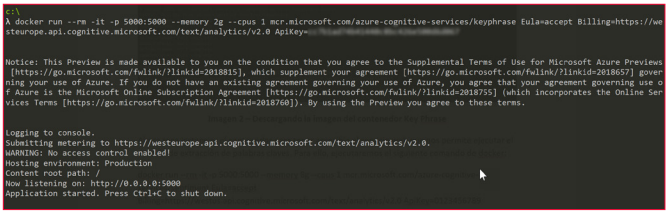

Imagen 3.– Ejecutando una instancia del contenedor Key Phrase.

Importante el parámetro Eula, Billing y ApiKey, que nos permiten ejecutar el servicio en el contenedor, pero con la información de facturación necesaria. Para esto, tenéis que crearos un servicio de Text Analytics creado en una de vuestras suscripciones de Azure de la que vamos a utilizar el Endpoint como valor del parámetro Billing y una de las Keys como valor del parámetro ApiKey.

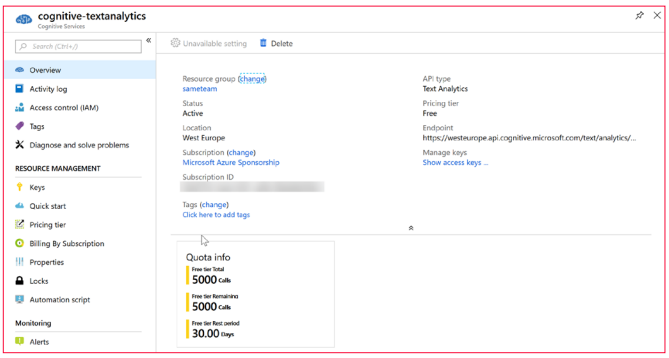

Imagen 4.– Azure Cognitive Services Text Analytics

Haciendo predicciones

Ya tenemos disponible el servicio de extracción de frases claves con toda su especificación, que se ejecuta en Azure. Aún así, si navegamos a la URL http://localhost:5000/swagger/ podemos ver la especificación OpenAPI con la descripción de las operaciones disponibles.

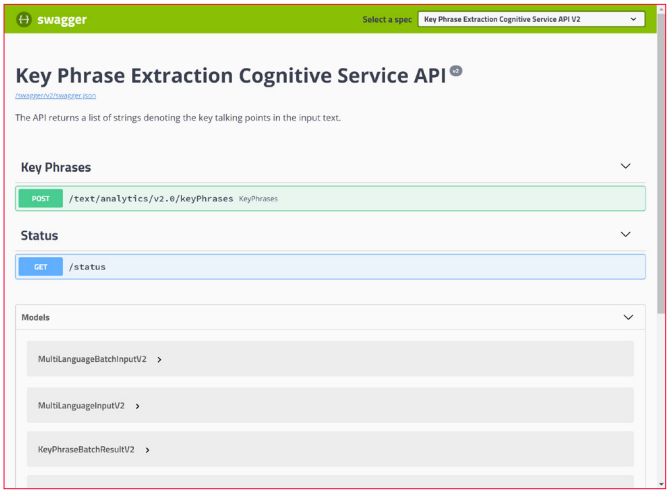

Imagen 5.– Definición OpenAPI del servicio

Probamos el servicio, haciendo una petición POST con el texto a analizar con el siguiente resultado:

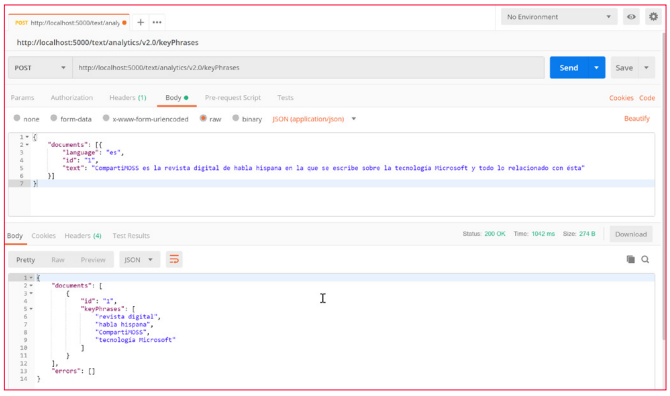

Imagen 6.– Definición OpenAPI del servicio.

Conclusiones

A partir de aquí, os toca experimentar con el resto de los servicios y ver cuando es necesario ejecutarlo en local, teniendo claro el abanico de escenarios posibles que esto nos habilita. Seguro que Microsoft ya está trabajando en publicar más de estos servicios como contenedores, aunque siempre nos queda preparar nuestro propio algoritmo, entrenarlo y publicarlo en nuestro contenedor de AI personalizado.

ALBERTO DIAZ MARTIN
MVP Azure
adiazcan@hotmail.com
@adiazcan
http://blogs.encamina.com/por-una-nube-sostenible/

35

SPFx Como cuidar el performance en nuestros desarrollos en ReactJS

SPFx ha traído una multitud de novedades para el desarrollador tradicional de SharePoint acostumbrado a un entorno más clásico y estable. Pero como los tiempos cambian hay que adaptarse a todo este tooling de herramientas (gulp, reactJS, typescript, yoeman, code, webpack…) que se han convertido en nuestras herramientas del día a día. Como en toda transición que se precie, hay un proceso que todo desarrollador debe conocer y ese no es otro que el momento en el que deja de utilizar una herramienta en modo normal y empieza a exprimirla al máximo y sacarla su máximo partido. En este articulo vamos a intentar explicar algunos detalles que hay que tener en cuenta cuando desarrollamos en ReactJS y que nos ayudan a mejorar el rendimiento de nuestra aplicación.

Partimos de la base de que para muchos de los desarrolladores de Office su primer contacto con ReactJS ha sido a través de SPFx. Para todos aquellos que se encuentren en esta situación les aconsejo que aprendan ReactJS independientemente de que después lo vayan a usar en SPFx o en otro tipo de desarrollo. Es vital conocer cómo funciona React, cuál es su ciclo de ejecución y todas las particularidades que tiene. De esta forma también viene muy bien para poder entender posteriormente como se ha montado SPFx. Muchas veces desde el desconocimiento pensamos que el éxito de esta librería es debido a la forma en la que trata el acceso al DOM y el algoritmo de Nodos que utiliza que no reemplaza todo el nodo sino el elemento que se ha modificado. Ahora bien, hay aspectos que no tenemos en cuenta y penalizan el rendimiento.

Evitar Actualizaciones del DOM innecesarias

El uso del “key”

Dentro de las listas del html el tener una Key en el mismo hace que los accesos al mismo sean mucho más rápidos, el no poner Key implica que para intentar añadir un nuevo elemento internamente se tenga que recorrer todos los elementos del mismo lo cual hace mucho más lento su acceso. Existen algunas herramientas como React Developer Tools que cuando estamos desarrollando nos marca en la consola que tenemos algunas etiquetas sin Key que intentemos solucionarlo. También existen reglas a la hora de compilación para evitar que pongamos la Key cuando toca (esta opción la aconsejo para evitar cometer errores tontos que después penalizan nuestra aplicación).

A la hora de poner una “key” también debemos tener en cuenta que esta sea única, es decir este código no sería correcto:

```
const List = ({items}) => {
  const listItems = items.map((item, index) => (
    <li key={index}>{item.name}</li>
  ))

  return (<ul>{listItems}</ul>)
}
```

Como vemos si tenemos una variable “index” que la ponemos a todos los <li> que tenemos es lo mismo que no tener nada. Tenemos que poner un código único por ejemplo dentro de la clase ítems podemos tener un Id, que es un elemento único y lo podemos utilizar como “key”

```
const List = ({items}) => {
  const listItems = items.map((item) => (
    <li key={item.id}>{item.name}</li>
  ))

  return (<ul>{listItems}</ul>)
}
```

Uso de High Order Component

Un High Order Component a partir de este momento abreviado como “HoC” es un patrón que se utiliza mucho en React para poder reaprovechar y reutilizar mucho de nuestros componentes. Este patrón lo podemos comparar como las muñecas Matriuskas en las que dentro de una muñeca hay otra más pequeña y así sucesivamente. Podemos hacer HoC por ejemplo para aplicar Seguridad a un componente o por ejemplo añadir un título a nuestro componente. Un ejemplo de High Order Component puede ser el siguiente

```
const withTitle = (Component) => (
  ({title, ...props}) => (
    <div>
      <h3>{title}</h3>
      <Component {...props} />
    </div>
  )
)
```

“SPFx ha traído una multitud de novedades para el desarrollador tradicional de SharePoint acostumbrado”

¿Que problema podemos tener con este HoC? Pues si este artefacto cuando vamos a hacer uso de él lo hacemos en el método Render del componente que se está implementando, implica que cada que ves que vuelve a pintar este componente estamos creando un nuevo componente. La solución correcta seria extraer este HoC a una variable más global e invocarla cuando vayamos a usarlo.

Ejemplo

```
class Emails extends React.Component {
  render() {
    const ListWithTitle = withTitle(List)
    const {emails} = this.props

    return (
      <section>
        <header> ... </header>
        <ListWithTitle items={emails} title="Emails" />
        <footer> ... </footer>
      </section>
    )
  }
}
```

Sino que deberíamos sacar la inicialización del componente fuera del render incluso fuera de la propia clase

```
const ListWithTitle = withTitle(List)

class Emails extends React.Component {
  render() {
    const {emails} = props

    return (
      <section>
        <header> ... </header>
        <ListWithTitle items={emails} title="Emails" />
        <footer> ... </footer>
      </section>
    )
  }
}
```

Evita renderizar los componentes de forma innecesaria

Como he comentado anteriormente, es necesario que tengamos claro cuál es ciclo de vida que tiene cada componente de React y que métodos son donde están y en qué momento se ejecutan: ComponentDidUpdate, ComponentWillMount, ShouldComponentUpdate, etc. También debemos tener en cuenta que hay algunas reglas no escrita que no debemos de saltar, por ejemplo, no debemos de modificar el state cada vez que el componente se esté actualizando.

"una Key en el mismo hace que los accesos al mismo sean mucho más rápidos"

Junto con este tipo de reglas debemos tener en cuenta que hay un método ShouldComponentUpdate. Este método devuelve un valor booleano indicando si tenemos que volver a pintar el componente o no. Sabemos que cada componente de ReactJS se actualiza siempre que se modifica bien el State o las Props, ahora bien, en algunas ocasiones solo queremos volver a pintar el componente si hay algo que cambia su valor bien en el state o bien en las propos, por lo que es recomendable crearse esta función e indicar en qué momento vamos a querer que el componente se reenderize. Por ejemplo.

```
class DataTable extends React.Component {
  shouldComponentUpdate(nextProps, nextState) {
    const {props, state} = this

    return !shallowEqual(props, nextProps)
      || !shallowEqual(state, nextState)
  }

  render() {
    // only gets called if next props OR state
    // don't shallow equal previous versions
  }
}
```

En este caso vamos a utilizar una librería shallowEqual que se encarga de comparar si los dos objetos que se pasan son igual o diferente.

Divide y vencerás: Separa el html en varios componentes

Uno de los fallos princípiales es que estamos acostumbrados a no pensar en componentes sino en un único punto donde tenemos todo el html. Este error es muy común porque nuestro pensamiento es pensar que no todo son Componentes y vemos el html como un único componente. Aparte de que esto no es correcto desde el punto de vista del mantenimiento del código fuente, desde el punto de vista solo de React hay un problema de rendimiento ya que al tener todo el desarrollo en un único componente hace que cualquier modificación en el mismo se tenga que "repintar" todo el componente y no solo una parte de él. Lo que hace que sobrecarguemos los accesos al DOM. Pensar en un componente como este:

```
class Page extends React.Component{
return (
    <main>
        <nav className="nav">
        <div className='navbar navbar-inverse'>
        <div className='navbar-header'>
            <button type='button' className='navbar-toggle' data-toggle='collapse' data-target='.navbar-collapse'>
                <span className='sr-only'>Toggle navigation</span>
                <span className='icon-bar'></span>
                <span className='icon-bar'></span>
                <span className='icon-bar'></span>
            </button>
            <Link className='navbar-brand' to={'/'}>Gestión Cartera</Link>
        </div>
        <div className='clearfix'></div>
        <div className='navbar-collapse collapse'>
            <ul className='nav navbar-nav'>
                <li>
              <NavLink to={'/assignemployee'} activeClassName='active'>
                        <span className='glyphicon glyphicon-fire'></span> Asignar Empleados
                    </NavLink>
                </li>
                <li>
                    <NavLink to={'/week'} activeClassName='active'>
                        <span className='glyphicon glyphicon-calendar'></span> Semanas
                    </NavLink>
                </li>
            </ul>
        </div>
    </div>
    <SearchForm query={this.state.query} onChange={this.handelChange} />
    <section className='data-table'>
    <MarqueeSelection selection={this._selection}>
        <DetailsList
            items={items}
            compact={isCompactMode}
            columns={columns}
            selectionMode={this.props.isModalSelection ? SelectionMode.multiple : SelectionMode.none}
            setKey="set"
            layoutMode={DetailsListLayoutMode.justified}
            isHeaderVisible={true}
            selection={this._selection}
            selectionPreservedOnEmptyClick={true}
            enterModalSelectionOnTouch={true}
            ariaLabelForSelectionColumn="Toggle selection"
            ariaLabelForSelectAllCheckbox="Toggle selection for all items"
        />
    </MarqueeSelection>
    </section>
        </nav>
    </main>
)
}
```

Cada vez que se produce un cambio en la búsqueda está pintando la tabla y la paginación lo que hace que estemos actualizando constantemente el DOM. Una opción mucho

mejor sería separar cada parte en componentes como el siguiente:

```
import DataTable from './DataTable';

class Nav extends React.PureComponent { ... }
class Pagination extends React.PureComponent { ... }

class Page extends React.Component {
  return (
    <main>
      <Nav navItems={NAV_ITEMS} />

      <SearchForm query={this.state.query} onChange={this._handleChange} />

      <DataTable rows={this.props.data} />

      <Pagination theme="blue" />
    </main>
  )
}
```

Cuando trabajamos con Redux que aspectos debemos de tener en cuenta.

En artículos anteriores hablé sobre la utilización del patrón Flux dentro de nuestros desarrollos en React, para ello utilizábamos algunas de las librerías que se encargan de abstraernos de este patrón, como pueda ser Redux, React-Redux, etc.. ¿Qué aspectos debemos tener en cuenta cuando usamos este patrón? Pues hay acciones que las estamos duplicando cuando realmente estamos haciendo lo mismo, lo ideal sería unificar estas dos acciones en una única acción y de esta forma evitamos modificar dos veces el state de nuestra aplicación y por lo tanto tendrá un rendimiento óptimo. Veamos un ejemplo:

Dado este contenedor

```
const getVisibleTodos = (todos, filter) => {
  // filters `todos` array by `filter` string
}
const mapStateToProps = (state) => ({
  todos: getVisibleTodos(state.todos, state.filter)
})

export default connect(mapStateToProps)(TodoList)
```

Existe una librería “reselect” que se encarga de unir todas estas propiedades con tal de unificar las llamadas al state global de nuestra aplicación. Quedando la aplicación de la siguiente forma:

```
import {createSelector} from 'reselect';

const getTodos = (state) => state.todos
const getVisibilityFilter = (state) => state.filter

const getVisibleTodos = createSelector(
  [getTodos, getVisibilityFilter],
  (todos, filter) => {
    // filters `todos` array by `filter` string
  }
)
const mapStateToProps = (state) => ({
  todos: getVisibleTodos(state)
})
```

Rendimiento custom de SPFx

Cuando trabajamos con SPFx tenemos que darnos cuenta de cuál es la estructura que ha montado el equipo de SharePoint para poder hacer WebParts utilizando JavaScript. Antes de empezar con SPFx debemos tener en cuenta cómo funciona la solución y en qué momento empieza a cargar nuestro desarrollo. Como sabemos nuestro WebPart hereda de la siguiente clase BaseClientSideWebPart, esta clase tiene muchos métodos y funcionalidad básica (alguna de la cual no está documenta). ¿Como podemos evitar algunos errores que cometemos de base? Pues de una forma sencilla, lo primero que tenemos que saber es si queremos volver a renderizar el componente o no. Existe un método “RenderOnce” que nos indica si el componente ya está renderizado o no. Por ello en nuestro desarrollo debemos de tener en cuenta de que si solamente queremos volver a renderizar nuestro componente una única vez hasta que recarguemos la página tendremos que invocar a este método. Por otro lado, hay que tener en cuenta de que, si nuestro WebPart tiene algún elemento configurable en la Toolpart y dependiendo de este valor puede cambiar su información, quizás no es conveniente que añadamos este control.

Conclusiones

A pesar de que React sea una librería con una historia muy pequeña si la comparamos con otras librerías hay que muchos aspectos que no utilizamos en nuestro día a día y que pueden mejorar el rendimiento de nuestra aplicación. En ocasiones las características propias de React hacen que se camuflen errores en el desarrollo.

Los desarrolladores de SharePoint estamos acostumbrados a que las APIs de SharePoint no son lo más rápidas del mundo y muchos de nuestros desarrollos cuando los analizamos por primera vez decimos que son culpa de SharePoint y del acceso a estas API. Sin embargo, como en la vida todo es mejorable y todo código se puede mejorar y dado que nuestro conocimiento con React es algo “nuevo” debemos de leer como poder mejorar este rendimiento y por analizar nuestro desarrollo y no echar las culpas una y otra vez a SharePoint.

Bibliografía

https://evilmartians.com/chronicles/optimizing-react-virtual-dom-explained

https://building.calibreapp.com/debugging-react-performance-with-react-16-and-chrome-devtools-c90698a522ad

https://reactjs.org/blog/2018/09/10/introducing-the-react-profiler.html

http://www.benmvp.com/slides/2018/reactalicante/react-perf.html#/

ADRIÁN DIAZ CERVERA
Software Architect Lead at Encamina
MVP Office Development
http://blogs.encamina.com/desarrollandosobresharepoint
http://geeks.ms/blogs/adiazcervera
adiaz@encamina.com @AdrianDiaz81

38

Microsoft Teams y SharePoint Online, el matrimonio perfecto

SharePoint Online (SPO) es un elemento fundamental en Microsoft Teams que proporciona no solo capacidades de almacenamiento de archivos y gestión documental, sino también características de integración que permiten que cualquier usuario pueda visualizar e interactuar directamente en Teams con páginas de SPO, listas o aplicaciones de la misma forma que lo harían desde un sitio de SPO. En este artículo vamos a revisar las posibilidades de integración entre ambas plataformas.

SPO como elemento fundamental de Microsoft Teams

SPO es uno de los bloques claves de la arquitectura de Microsoft Teams proporcionando funcionalidad de almacenamiento de documentos para los canales que forman parte de un equipo de Microsoft Teams. Cuando se crea un Team de Microsoft Teams, se crea un sitio moderno de SPO que proporciona estas características.

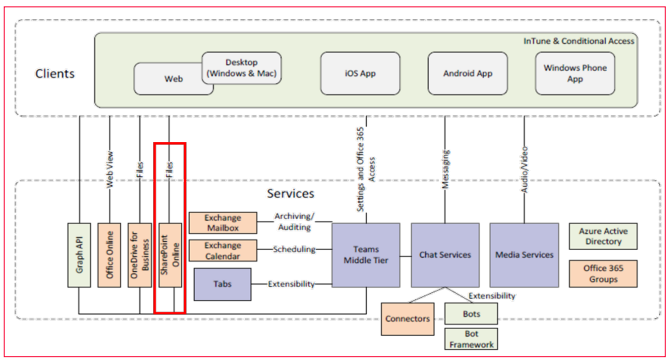

Imagen 1.- Arquitectura de Microsoft Teams.

Además de la integración nativa de SPO en Microsoft Teams, es posible integrar ambas plataformas de forma directa a través de los elementos que se indican a continuación:

- Pestañas, que permiten añadir acceso a páginas de SPO y listas en canales de Microsoft Teams.
- Conectores, que permiten recibir notificaciones de acciones ocurridas en SPO (Por ejemplo, noticias publicadas) en canales de Microsoft Teams.
- Cloud Storage, que permite añadir en la pestaña "Archivos" un acceso a otra biblioteca de documentos de nuestro tenant de SPO.
- Flow, a través de la creación de pequeños procesos que permitan por ejemplo enviar mensajes a un canal de Microsoft Teams cuando se crea un documento o un elemento de lista en un sitio de SPO.
- PowerApps, a través de añadir una PowerApp a un canal de Teams que facilite interactuar con datos de SPO.
- Las pestañas de OneNote y Wiki que permiten tomar notas en canales de Teams y que se almacenan en el sitio asociado al Team.

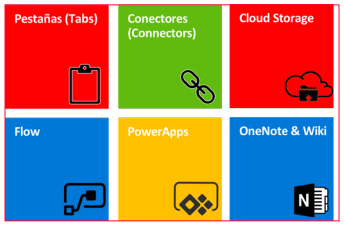

Imagen 2.- Posibilidades de integración entre Microsoft Teams y SPO.

Integración de SPO y Microsoft Teams por medio de pestañas

La primera posibilidad más clara de integrar ambas plataformas es a través de pestañas que permiten agregar de forma nativa en Teams el acceso a elementos de SPO como páginas, listas o bibliotecas de documentos. En el momento de redacción de este artículo, disponemos de las siguientes pestañas para añadir páginas, listas o bibliotecas en canales de Teams:

- Biblioteca de documentos, permite añadir una biblioteca de documentos al canal de Microsoft Teams. Se puede seleccionar una biblioteca de un sitio relevante (esta información es provista por el Graph) al que tiene acceso el usuario o bien agregando directamente el enlace a la biblioteca de documentos.
- SharePoint, permite añadir a un canal de Microsoft Teams una página o una lista del sitio asociado al Team.
- Sitio web, permite añadir cualquier página, lista o biblioteca de documentos a un canal de Microsoft Teams.

Imagen 3.- Pestañas para añadir páginas, listas y/o bibliotecas de documentos en canales de Microsoft Teams.

Si hacemos uso de la pestaña Biblioteca de documentos, el resultado tras seleccionar una biblioteca de documentos a la que tengamos acceso es el siguiente:

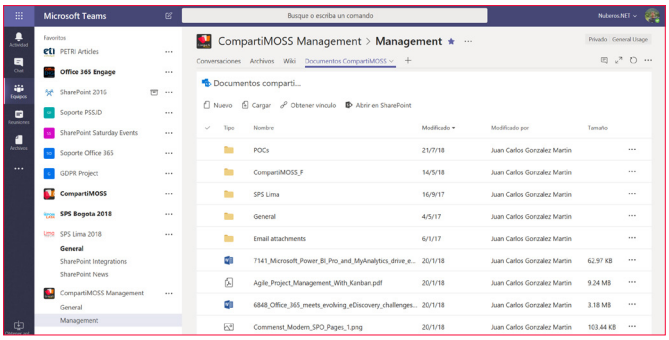

Imagen 4.- Resultado que se obtiene al utilizar pestaña Biblioteca de documentos.

Como se puede apreciar, la pestaña muestra el contenido de la biblioteca de documentos que se ha añadido, aunque no se muestran la barra de opciones nativa de bibliotecas de documentos o los metadatos que se hayan podido configurar (*Nota: Esta es una mejora que está previsto que sea liberada por Microsoft antes de que acabe el año*).

En cambio, la pestaña SharePoint nos permite agregar cualquier página o lista del sitio asociado al Team y el resultado que se obtiene es el siguiente:

- Visualización de una página moderna en Teams: La página se muestra de la misma forma que si se accede a esta desde la interfaz de usuario de SPO.

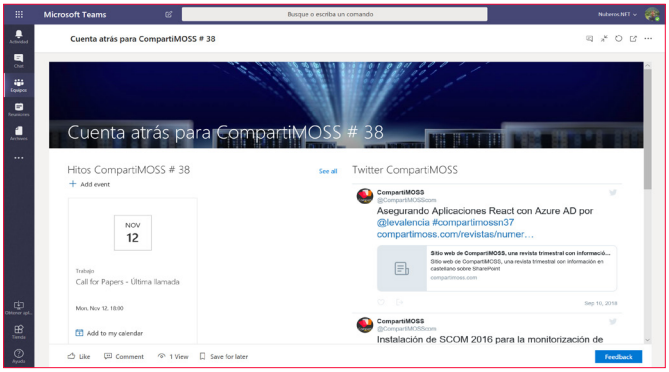

Imagen 5.- Visualización de una página de SPO en Microsoft Teams.

- Visualización de una lista del sitio en SPO: La experiencia de visualización de la lista en Teams es similar a la que tenemos en la interfaz de usuario de SPO con la diferencia de que la barra de acciones no dispone de todas las acciones disponibles. Si se ha aplicado formato a la lista haciendo uso del formato de columnas y/o formato de vistas, la personalización se muestra también en Teams.

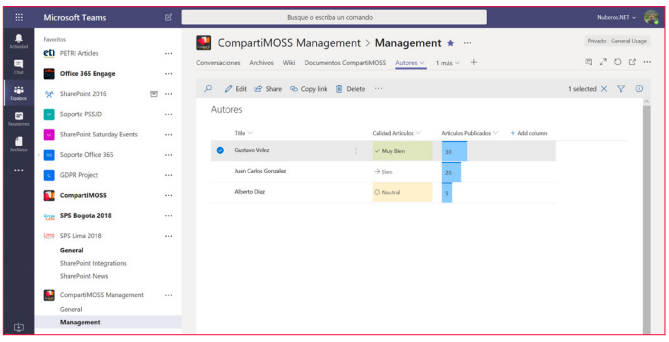

Imagen 6.- Visualización de una lista en Microsoft Teams.

Finalmente, la pestaña Sitio web permite visualizar una página, una lista o una biblioteca de documentos en SPO en Microsoft Teams de la misma forma que con las pestañas Biblioteca de documentos y SharePoint.

Integración mediante Almacenamiento en la nube

Desde la pestaña "Archivos" de un canal de Microsoft Teams es posible añadir el acceso a cualquier biblioteca de documentos de nuestro tenant de SPO haciendo uso de la opción "Almacenamiento en la nube".

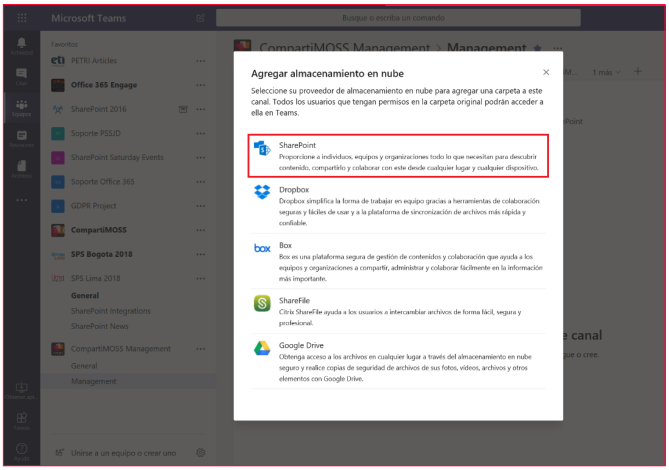

Imagen 7.- Agregar almacenamiento en nube a un canal de Teams.

Tras hacer clic en SharePoint, podremos seleccionar el acceso a la Biblioteca de documentos que se desea añadir bien desde la lista de Sitios sugeridos que se presenta o bien añadiendo manualmente la URL de la biblioteca. El resultado que se obtiene es un acceso directo a la biblioteca que se muestra como una carpeta más del canal.

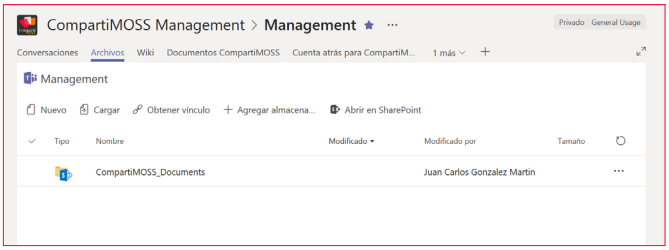

Imagen 8.- Acceso a la biblioteca de documentos añadida a través de "Agregar almacenamiento en nube".

Integración mediante Conectores

Los conectores en Microsoft Teams habilitan integrar datos procedentes de fuentes o servicios externos en conversaciones en canales de Microsoft Teams. Una de esas fuentes externas puede ser SharePoint y en concreto, disponemos por defecto de un Conector de noticias que permite notificar, una vez se ha configurado, que se ha publicado una nueva noticia en un sitio de SharePoint. Para añadir el Conector de Noticias de SharePoint:

- Hacemos clic en los "…" al lado del nombre del canal donde queremos añadir el conector. A continuación, hacemos clic en "Conectores" para que se muestre el diálogo de selección de conector en el que buscaremos

el conector de “Noticias de SharePoint”.

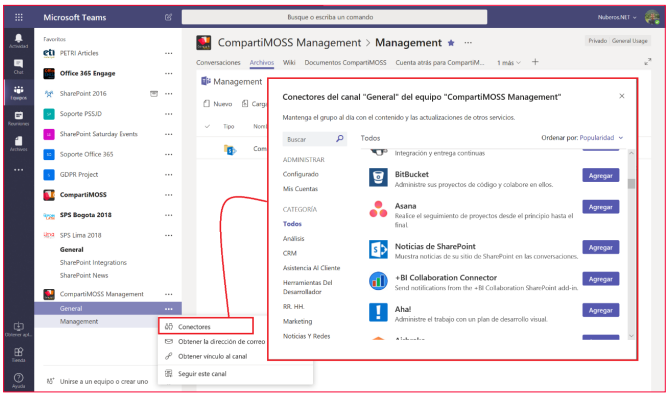

Imagen 9.- Agregando un conector a un canal de Microsoft Teams.

- Hacemos clic en el botón “Agregar” y a continuación, en la nueva ventana que se muestra haremos clic en “Instalar” para que se añada el conector al canal. Para finalizar, haremos clic en “Guardar” (*Nota: Es posible que la primera vez que añadáis el conector de “Noticias de SharePoint” os de un error tras seguir estos pasos, por lo que tendréis que volver a seguir los pasos para añadir el conector con la diferencia de que solo será necesario realizar el paso de “Configurar”*).
- Una vez configurado el conector, se debería ver en el canal un mensaje indicando que se ha agregado al mismo y que el conector está “contectado”.

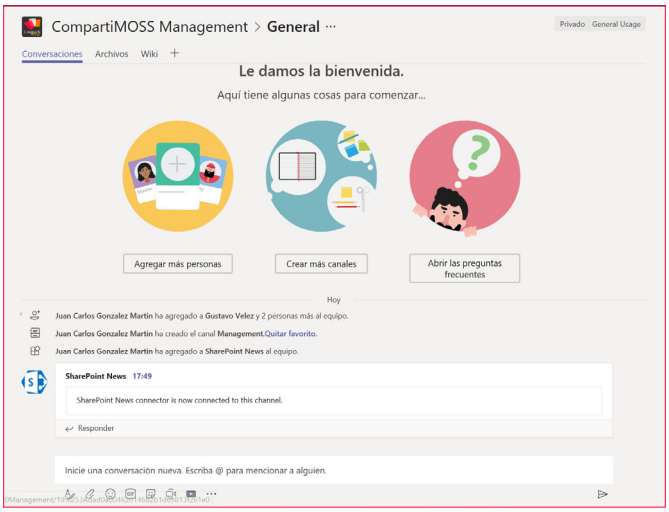

Imagen 10.- Mensaje que se añade al canal una vez se ha configurado el conector correctamente.

- Finalmente, para probar el conector simplemente publicamos una noticia en el sitio y comprobamos que el conector añade un mensaje relativo a la misma en el canal.

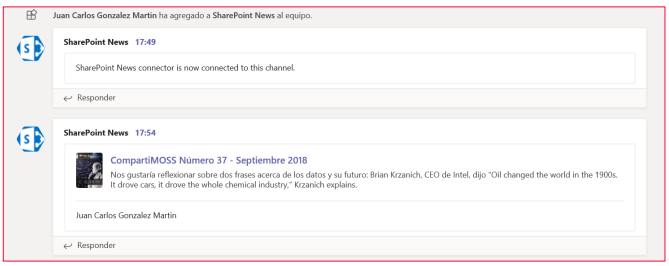

Imagen 11.- Mensaje añadido por el conector cuando se publica una noticia.

Integración con PowerApps y Flow

Para finalizar el artículo, veremos la integración de Microsoft Teams y SPO por medio de PowerApps y de Flow. En el caso de PowerApps, se puede añadir cualquier PowerApp como una pestaña de Microsoft Teams lo que incluye una PowerApp que se haya creado a partir de una lista de SPO.

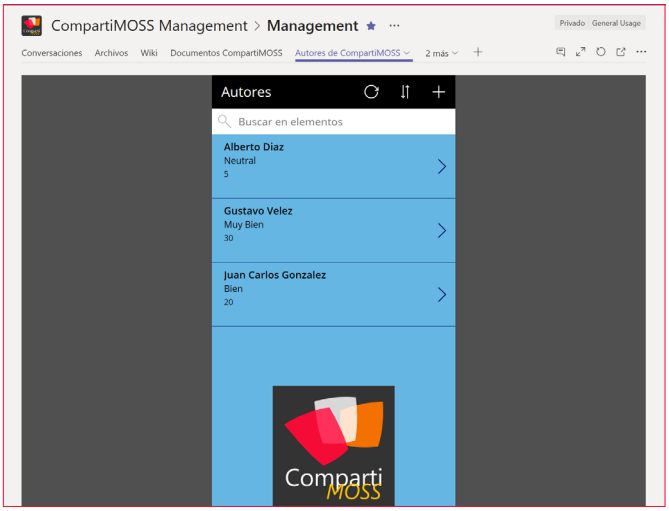

Imagen 12.- Integración de Teams y SPO por medio de PowerApps.

Finalmente, en el caso de Flow disponemos de plantillas predefinidas que hacen uso de acciones de SPO y Teams o bien podemos crear un Flow desde cero. A modo de ejemplo, la siguiente imagen muestra las plantillas de Flow disponibles por defecto para Microsoft Teams algunas de las cuales incluyen acciones con SPO.

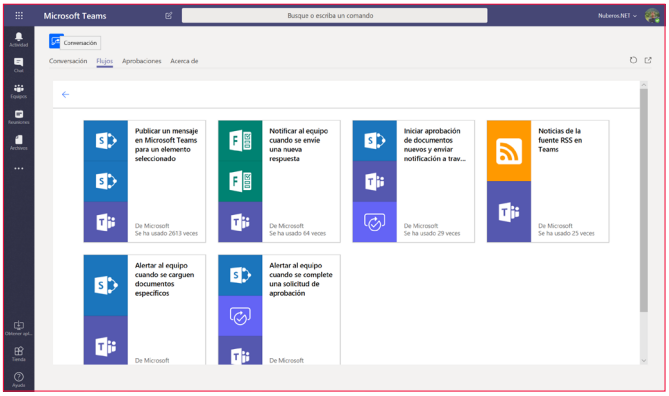

Imagen 13.- Plantillas por defecto de Microsoft Teams en la App de Flow.

Conclusiones

SharePoint Online no solo es una pieza clave de Microsoft Teams, sino que también se puede integrar con la plataforma por medio de distintas posibilidades: pestañas, almacenamiento nube, conectores, Flows de Microsoft Flow o PowerApps. En el futuro, Microsoft seguirá mejorando las características de integración a la vez que habilitará que los desarrolladores puedan construir aplicaciones que muestren información de SPO en Teams y a la inversa.

JUAN CARLOS GONZÁLEZ
Cloud & Productivity Advisor
Office Apps & Services MVP
jcgm1978
https://jcgonzalezmartin.wordpress.com/

41

Get-CsLatam, La primera conferencia de Skype y Teams en español

Todos los hispanoparlantes nos hemos dado cuenta de que tenemos conferencias de calidad siempre en idioma extranjero, y aunque nosotros podemos hacer nuestros propios eventos, no son al 100% especializados en un tema en particular, y tenemos que escuchar los speech de la gente de ventas y marketing.

Con eso en mente, es que cuatro individuos deciden realizar la primera conferencia dirigida por la comunidad, y enfocada única y exclusivamente a Skype for Business y Microsoft Teams. Es por ello que nace Get-CsLatam. Conferencia que en su primera edición se realizó en la ciudad de México, teniendo más de 100 asistentes presenciales y 300 remotos. Contando con once speakers de calidad, no solo de la ciudad de México, sino también contamos con presencia de Guadalajara, Mexicali, y desde Bogotá y Medellín en Colombia.

Más de 6 meses de planeación, nos llevaron a tener una conferencia de calidad. Una conferencia como la que desde hace mucho tiempo se merecía la comunidad de Skype y Teams en español. Fue una larga jornada, pero valió la pena.

Al hacer el Call for Speakers, tuvimos que rechazar a algunos, no porque su contenido no fuera el que buscábamos, sino que por ser nuestra primera edición no contábamos con tanto patrocinio como para traer gente de otros países. Cada uno de nuestros speakers hicieron el esfuerzo por buscar los recursos para venir hasta la ciudad de México y presentar su sesión. De antemano agradezco a todos y cada uno de ellos.

¿A quién tuvimos como speaker? Aquí la lista y los temas que cada uno de ellos presento. (Ordenados de acuerdo al horario de su charla)

Leonardo Cruz - Arquitecto Tecnológico para Socios para Latinoamérica en Microsoft. Fue el encargado de abrir el evento, fue nuestro Keynote speaker. Con su charla de Modern Workplace con MS Teams, fue quien abrió el espacio para todos los asistentes.

Vicente Guzmán - rMVP Windows Development, con su sesión creando Bots para Microsoft Teams. Toda la gente interesada en la creación de bot y desarrollo sin dudarlo asistió a la charla de Vicente, quien como lo vemos tiene experiencia siendo un rMVP.

Rodolfo Castro – MVP Office Apps and Services. Actualmente único MVP en LATAM totalmente enfocado a Skype y Teams. Brindo la charla Monitoring y QoS en Microsoft Teams, donde nos mostró que tan importante es poner etiquetado en la red, para dar una mejor experiencia al usuario.

Haaron González – MVP especialista en SharePoint. Nos visitó desde Mexicali, para brindarnos su charla SharePoint y Teams, creando aplicaciones empresariales. Actualizando su sesión un día antes, para que los asistentes tuvieran la información más reciente, Haaron cautivo a la audiencia developer con su charla.

Christian Romano – rMVP Office Server and Services. Christian además de speaker también es uno de los organizadores del evento, la charla que nos dio fue Troubleshooting MS Teams, vimos las herramientas necesarias para resolver problemas con Teams y cuando ocuparlas.

> ***" los hispanoparlantes nos hemos dado cuenta de que tenemos conferencias de calidad siempre en idioma extranjero"***

Geovany Acevedo – rMVP Office Server and Services, Geovany ha estado trabajando con la parte de adopción, governance, security y compliance para su empresa, para poder adoptar de la mejor manera MS Teams. Toda esa experiencia nos la transmitió en su sesión "Security and Compliance en MS Teams"

Fernando Chiquiza – MVP Office Apps and Services, viajando desde su natal Bogotá en Colombia, nos apoyó con la participación de su charla "Teams como ambiente colaborativo", con su manera peculiar de impartir su sesión,

Fernando tuvo a la audiencia siempre atenta a su sesión, y aprendieron mejor como funciona Teams, SharePoint y OneDrive.

Jose Luis Arroyo – Líder UC and O365 Product Owner para una entidad de gobierno, Jose Luis es el único de los speakers que tiene la visión del cliente, él no es parte de Microsoft, ni de algún partner o vendor. El sí es un cliente final, el cual ve los pros y los contras desde su perspectiva. Fue un placer verlo en su charla "Transformación Digital y Productividad Dia 0"

"Para finalizar el evento tuvimos una pequeña happy hour en un restaurante cercano"

Juan Camilo Martinez – Nuestro speaker paisa, traído desde la hermosa ciudad de Medellín en Colombia. Camilo tiene la experiencia de muchos años en Comunicaciones Unificadas de Microsoft, trabajando para uno de los Partners más grandes en Colombia y Latam, nos hace el honor de mostrarnos su charla "Voice Routing en MS Teams". Camilo fue uno de los organizadores del evento, aun estando remoto fue de gran ayuda a la realización del evento.

Jose Roberto Correa – Ingeniero en Sistemas, Gerente de IT en sector gobierno. Otro de los organizadores de este gran evento, nos guio en el camino para movernos de Skype a Microsoft Teams con su charla "Adopción en Microsoft Teams" donde pudimos ver, donde ha estado Microsoft con anterioridad, donde está ahora, y a donde podemos llegar. Excelente sesión para cerrar.

El evento abrió sus puertas a las 8 am para los asistentes, donde pudieron convivir con la comunidad, hacer networking y ver a nuestros patrocinadores. Tuvieron un rico desayuno con comida típica mexicana.

Frijoles, Chilaquiles, chile poblano, y huevo.

También nuestros patrocinadores son dignos de reconocer, ya que confiaron en este evento, siendo el primero que se hace, y ellos nos apoyaron en todo momento. Gracias Polycom, Plantronics, Audiocodes y C3ntro Telecom.

Para finalizar el evento tuvimos una pequeña happy hour en un restaurante cercano, donde igualmente tuvimos una excelente cena rodeada de antojitos mexicanos como sopes, tacos de lengua, arrachera, cochinita, huaraches, frijoles y cervezas.

El evento fue un éxito de inicio a fin, gracias a los speakers, patrocinadores y a los asistentes en general.

Pueden ver más del evento en https://get-cslatam.com, o en sus redes sociales https://twitter.com/getcslatam .

Les dejo con algunas fotos del evento, pero pueden buscar el hashtag #GetCsLatam donde podrán encontrar todo lo que compartió la comunidad ese día. Espero que con las fotos se motiven y ¡esperamos contar con la presencia de ustedes estimados lectores para GetCsLatam 2019!

Esten pendientes

RODOLFO CASTRO AGUILAR
MVP Office Apps and Services
Twitter : @ucblogmx
facebook.com/groups/SkypeTeamsUG/
ucblogmx.com

43 Introducción a Office 365 Management Activity API

Conceptos generales

Office 365 Management Activity API proporciona una serie de operaciones que permiten acceder a la información de actividad de un tenant de Office 365. Esta información se almacena en conjunto de Content BLOBs, de los cuales podemos extraer la información que nos interesa, tal y como se muestra en la siguiente figura:

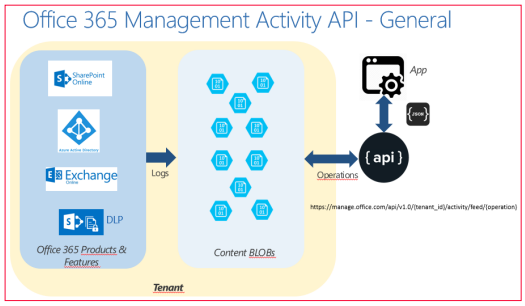

Imagen 1.- Componentes de la API.

Como se puede observar, existen hasta 4 tipos de contenido que se pueden consultar:

- Audit.AzureActiveDirectory: Actividades relacionadas con Azure Active Directory, como puede ser el login o el logout de los usuarios.
- Audit.Exchange: Actividades relacionadas con Exchange, como puede ser el envío o la recepción de correos electrónicos.
- Audit.Sharepoint: Actividades relacionadas en SharePoint, como pueden ser la visualización de una página o la descarga de un fichero.
- Audit.General: Actividades que no se incluyen en los tipos de contenido previos.
- DLP.All: Actividades relacionadas con las políticas de Data Loss Prevention, como por ejemplo el intento de compartición de ficheros con usuarios no autorizados.

Es importante notar que desde que se produce una actividad hasta que ésta se almacena en un Content BLOB pueden pasar hasta 24h, con lo que la información almacenada en éstos no tiene por qué estar ordenada cronológicamente.

"detectar alguna acción inadecuada, o simplemente monitorizar en qué fecha o desde que equipo se llevó a cabo"

Operaciones

Para poder empezar a trabajar con la API necesitamos saber las operaciones actualmente existentes, y que se presentan a continuación:

- Iniciar una suscripción, de forma que podamos empezar a guardar la información del registro de auditoría de un tipo de contenido definido en los Content BLOBs.
- Parar una suscripción, para dejar de guardar la información del registro de auditoría del tipo de contenido seleccionado.
- Obtener una lista de las suscripciones actuales, ya que podemos tener distintas suscripciones (una por tipo de contenido).
- Obtener la lista de contenido disponible, mediante la cual podremos consultar la información almacenada en los distintos Content BLOBs (mediante unas URLs).
- Recibir notificaciones cuando existe nuevo contenido disponible para la consulta y mediante el uso de un webhook.
- Ver el contenido que hay dentro de un Content BLOB (mediante su URL), con la información concreta según el tipo de contenido de la suscripción.
- Obtener una lista de notificaciones enviadas por un webhook.
- Obtener los nombres descriptivos de un recurso mediante su GUID.

Un ejemplo práctico

Seguidamente presentaremos un ejemplo de cómo usar algunas de las operaciones introducidas anteriormente. En primer lugar, deberíamos iniciar una suscripción:

Iniciar una suscripción (POST)

```
https://manage.office.com/api/v1.0/[redacted]-41f5-94d5-65047663d8c4/activity/feed/subscriptions/start?contentType=Audit.Sharepoint&publisherIdentifier=[redacted]-51fa-92d5-65047853e7c3
```

Imagen 2.- Operación para iniciar una suscripción.

Donde el primer parámetro corresponde al identificador del tenant en el que queremos iniciar la operación, mientras que el publisherIdentifier es un identificador que sirve para controlar quien está haciendo la operación, y que sirve para controlar el número de llamadas realizadas a la API (actualmente el límite se encuentra en 60K por minuto). Si no se indica este parámetro, se comparte la cuota con las otras peticiones que tampoco lo indiquen.

A continuación, podríamos consultar las suscripciones actuales:

Obtener las suscripciones actuales (GET)

```
https://manage.office.com/api/v1.0/[redacted]-41f5-94d5-65047663d8c4/activity/feed/subscriptions/list
```

Imagen 3.- Operación para obtener la lista de suscripciones actuales.

Como en todas las operaciones, necesitaremos el identificador del tenant para poder ejecutar la operación.

Un tercer paso posible sería la obtención de los Content BLOBs disponibles, tal y como se muestra en la siguiente imagen:

Obtener la lista de Content Blobs disponibles (GET)

```
https://manage.office.com/api/v1.0/[redacted]-41f5-94d5-65047663d8c4/activity/feed/subscriptions/Content?ContentType=Audit.Sharepoint
```

Imagen 4.- Operación para obtener la lista de Content BLOBs disponibles.

> Como en casos anteriores, en la URL de la petición se indica el identificador del tenant, así como el tipo de contenido sobre el que queremos realizar la consulta.

Finalmente, podríamos ver el contenido de un Content BLOB mediante la siguiente petición:

Obtener el detalle de un Content Blob (GET)

```
https://manage.office.com/api/v1.0/[redacted]-41f5-94d5-65047663d8c4/activity/Details/20180606071654778031970$20180606071654778031970$audit_sharepoint$Audit_SharePoint
```

Imagen 5.- Operación para obtener el contenido de un Content BLOB

> En este caso, a parte del identificador del tenant también necesitamos el identificador del Content BLOB que queremos consultar, y que hemos obtenido con la operación de obtener la lista de contenido disponible.

El resultado obtenido de esta operación, y que es lo que realmente nos importa obtener al final de todo el proceso sería algo como lo que muestra la imagen siguiente:


```
{
    "CreationTime": "2018-05-21T08:40:22",
    "Id": "d662d905-ab87-48fe-80fe-08d5bef67dd3",
    "Operation": "FileUploaded",
    "OrganizationId": "[redacted]",
    "RecordType": 6,
    "UserKey": "i:0h.f|membership|10037ffe8161327c@live.com",
    "UserType": 0,
    "Version": 1,
    "Workload": "OneDrive",
    "ClientIP": "80.30.113.163",
    "ObjectId": "https://ferranchopo-my.sharepoint.com/personal/ferran_ferranchopo_com/Documents/[redacted]/Temario Formación .NET.docx",
    "UserId": "ferran@ferranchopo.com",
    "CorrelationId": "8240699e-3011-5000-da17-557e461c6a59",
    "EventSource": "SharePoint",
    "ItemType": "File",
    "ListId": "2011d055-6794-4147-b4f6-84efd23029e4",
    "ListItemUniqueId": "bea1b96a-9fde-4e7b-ae91-a659598b5b7d",
    "Site": "f051022a-2fd2-4354-88b0-cbf3aff2d653",
    "UserAgent": "Microsoft Office Word 2014",
    "WebId": "63d40601-2f9f-49ec-a67f-1e39b918273c",
    "SourceFileExtension": "docx",
    "SiteUrl": "https://ferranchopo-my.sharepoint.com/personal/ferran_ferranchopo_com/",
    "SourceFileName": "Temario Formación .NET.docx",
    "SourceRelativeUrl": "[redacted]"
},
{
    "CreationTime": "2018-05-21T08:40:22",
    "Id": "3e1d7bf1-48e5-477a-07f0-08d5bef67dfc",
    "Operation": "FileModified",
    "OrganizationId": "[redacted]",
    "RecordType": 6,
    "UserKey": "i:0h.f|membership|10037ffe8161327c@live.com",
    "UserType": 0,
    "Version": 1,
    "Workload": "OneDrive",
    "ClientIP": "80.30.113.163",
    "ObjectId": "https://ferranchopo-my.sharepoint.com/personal/ferran_ferranchopo_com/Documents/[redacted]/Temario Formación .NET.docx",
    "UserId": "ferran@ferranchopo.com",
    "CorrelationId": "8240699e-3011-5000-da17-557e461c6a59",
    "EventSource": "SharePoint",
    "ItemType": "File",
    "ListId": "2011d055-6794-4147-b4f6-84efd23029e4",
    "ListItemUniqueId": "bea1b96a-9fde-4e7b-ae91-a659598b5b7d",
    "Site": "f051022a-2fd2-4354-88b0-cbf3aff2d653",
    "UserAgent": "Microsoft Office Word 2014",
    "WebId": "63d40601-2f9f-49ec-a67f-1e39b918273c",
    "SourceFileExtension": "docx",
    "SiteUrl": "https://ferranchopo-my.sharepoint.com/personal/ferran_ferranchopo_com/",
    "SourceFileName": "Temario Formación .NET.docx",
    "SourceRelativeUrl": "Documents/[redacted]"
},
```


Imagen 6.- Contenido de un Content BLOB.

En el detalle del resultado se puede observar como tenemos el resultado de dos operaciones (FileUplodaded y FileModified), así como el servicio utilizado (Workload: Onedrive), la dirección IP desde la que se realizó la acción, el nombre del fichero, el usuario y otros datos adicionales.

Es importante saber que existen diferentes esquemas según el tipo de contenido, con lo que la información obtenida difiere entre éstos. Finalmente, y para acabar con este apartado, es muy importante notar que los Content BLOBs solo están disponibles para la consulta durante 7 días desde su fecha de creación.

Seguridad

Es obvio que para poder trabajar con la API en cualquier tenant necesitamos tener permiso para acceder a estos recursos. Por ello, en primer lugar, deberemos dar de alta nuestra aplicación en Azure AD (de nuestro tenant, ¡y no del cliente que queramos monitorizar!), y que solicitará los siguientes permisos al tenant que queramos monitorizar:

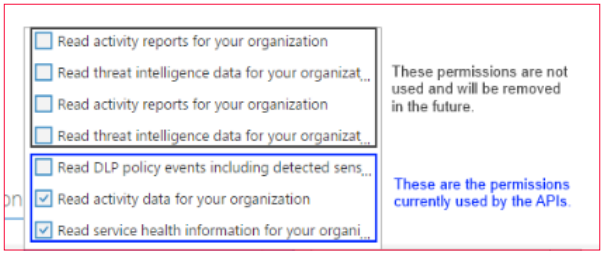

Imagen 7.- Permisos para poder usar la API en un tenant.

El segundo paso consistirá en obtener un token para poder realizar cualquiera de las operaciones presentadas anteriormente (deberá ser un parámetro más de la llamada). Para ello se pueden usar dos mecanismos: El flujo de concesión de código de autorización o el flujo de credenciales de cliente.

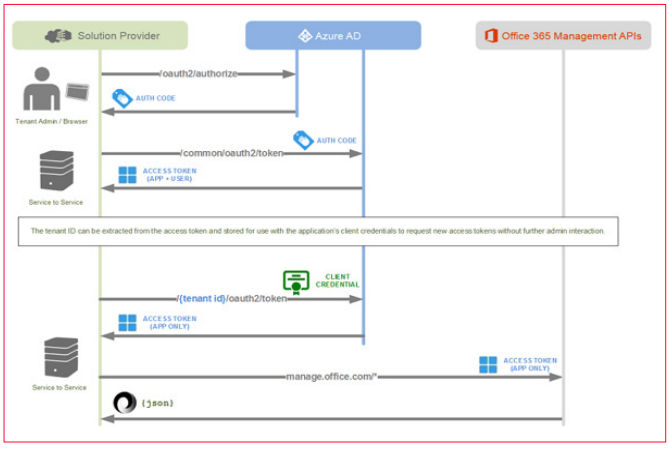

Imagen 8.- Proceso de autorización para poder usar la API

En nuestro caso elegimos la primera opción, con lo que deberemos hacer una petición como la siguiente:

```
GET
https://login.windows.net/common/oauth2/authorize?response_type=code&client_id=2d4d11a2-f814-46a7-890a-274a72a7309e&resource=https://manage.office.com&redirect_uri=https://www.miapp.net
```

En esta petición indicaremos el client_id de nuestra aplicación registrada previamente, así como la URL a la que se debe redirigir cuando se obtenga el código de autorización. De esta manera obtendremos un código de autorización, con el que haremos otra petición para obtener un token, que será el que usaremos para hacer las peticiones a la API, tal y como se muestra seguidamente:

```
POST common/oauth2/token HTTP/1.1

Host: https://login.windows.net

Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&client_id=2d4d11a2-f814-46a7-890a-
274a72a7309e&code=AwABAAAAvPM1KaPlrEqdFSBzjqfTGBCmLdgfSTLEMPGYuNHSUYBrqqf_Z
T_p5uEAEJJ_nZ3UmphWygRNy2C3jJ239gV_DBnZ2syeg95Ki-374WHUP-i3yIhv5i-
7KU2CEoPXwURQp6IVYMw-
DjAOzn7C3JCu5wpngXmbZKtJdWmiBzHpcO2aICJPu1KvJrDLDP2OchJBXzVYJtkfjviLNNW7l7Y3ydc
HDsBRKZc3GuMQanmcghXPyoDg41g8XbwPudVh7uCmUponBQpIhbuffFP_tbV8SNzsPoFz9CLpB
CZagJVXeqWoYMPe2dSsPiLO9Alf_YIe5zpi-
zY4C3aLw5g9at35eZTfNd0gBRpR5ojkMIcZZ6IgAA&redirect_uri=
https://www.miapp.net&client_secret=p@ssw0rd&resource=https://manage.office.com
```

Imagen 9.- Petición para obtener un token.

En este caso deberemos indicar el client_id y el client_secret obtenidos al registrar la aplicación en Azure AD (primer paso) y el código de autorización (obtenido en el anterior).

Finalmente obtendremos un token, con el que podremos hacer las llamadas a la API:

```
{
  "access_token": "
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Ik5HVEZ2ZEstZnl0aEV1THdqcHdBS
k9NOW4tQSJ9.eyJhdWQiOiJodHRwczovL3NlcnZpY2UuY29udG9zby5jb20vIiwiaXNzIjoia
HR0cHM6Ly9zdHMud2luZG93cy5uZXQvN2ZlODE0NDctZGE1Ny00Mzg1LWJlY2ItNmRlNTdmMj
E0NzdlLyIsImlhdCI6MTM4ODQ0MDg2MywibmJmIjoxMzg4NDQwODYzLCJleHAiOjEzODg0NDQ
3NjMsInZlciI6IjEuMCIsInRpZCI6IjdmZTgxNDQ3LWRhNTctNDM4NS1iZWNiLTZkZTU3ZjIx
NDc3ZSIsIm9pZCI6IjY4Mzg5YWUyLTYyZmEtNGIxOC05MWZlLTUzZGQxMDlkNzRmNSIsInVwb
iI6ImZyYW5rbUBjb250b3NvLmNvbSIsInVuaXF1ZV9uYW1lIjoiZnJhbmttQGNvbnRvc28uY2
9tIiwic3ViIjoiZGVOcUlqOUlPRTlQV0pXYkhzZnRYdDJFYWJQVmwwQ2o4UUFtZWZSTFY5OCI
sImZhbWlseV9uYW1lIjoiTWlsbGVyIiwiZ2l2ZW5fbmFtZSI6IkZyYW5rIiwiYXBwaWQiOiIy
ZDRkMTFhMi1mODE0LTQ2YTctODkwYS0yNzRhNzJhNzMwOWUiLCJhcHBpZGFjciI6IjAiLCJzY
3AiOiJ1c2VyX2ltcGVyc29uYXRpb24iLCJhY3IiOiIxIn0.JZw8jC0gptZxVC-
7l5sFkdnJgP3_tRjeQEPgUn28XctVe3QqmheLZw7QVZDPCyGycDWBaqy7FLpSekET_BftDkew
RhyHk9FW_KeEz0ch2c3i08NGNDbr6XYGVayNuSesYk5Aw_p3ICRlUV1bqEwk-
Jkzs9EEkQg4hbefqJS6yS1HoV_2EsEhpd_wCQpxK89WPs3hLYZETRJtG5kvCCEOvSHXmDE6eT
HGTnEgsIk--
UlPe275Dvou4gEAwLofhLDQbMSjnlV5VLsjimNBVcSRFShoxmQwBJR_b2011Y5IuD6St5zPnz
ruBbZYkGNurQK63TJPWmRd3mbJsGM0mf3CUQ",
  "token_type": "Bearer",
  "expires_in": "3600",
  "expires_on": "1388444763",
  "resource": "https://service.contoso.com/",
  "refresh_token":
"AwABAAAAvPM1KaPlrEqdFSBzjqfTGAMxZGUTdM0t4B4rTfgV29ghDOHRc2B-
C_hHeJaJICqjZ3mY2b_YNqmf9SoAylD1PycGCB90xzZeEDg6oBzOIPfYsbDWNf621pKo2Q3GG
THY1mNfwoc-
OlrxK69hkha2CF12azM_NYhgO668yfcUl4VBbiSHZyd1NVZG5QTIOcbObu3qnLutbpadZGAxq
jIbMkQ2bQS09fTrjMBtDE3D6kSMIodpCecoANon9b0LATkpitimVCrl-
NyfN3oyG4ZCWu18M9-vEou4Sq-1oMDzExgAf61noxzkNiaTecM-
Ve5cq6wHqYQjfV9DOz4lbceuYCAA",
  "scope": "user_impersonation",
"id_token": "
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJhdWQiOiIyZDRkMTFhMi1mODE0LTQ2YTctO
DkwYS0yNzRhNzJhNzMwOWUiLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC83ZmU4MT
Q0Ny1kYTU3LTQzODUtYmVjYi02ZGU1N2YyMTQ3N2UvIiwiaWF0IjoxMzg4NDQwODYzLCJuYmY
iOjEzODg0NDA4NjMsImV4cCI6MTM4ODQ0NDc2MywidmVyIjoiMS4wIiwidGlkIjoiN2ZlODE0
NDctZGE1Ny00Mzg1LWJlY2ItNmRlNTdmMjE0NzdlIiwib2lkIjoiNjgzODlhZTItNjJmYS00Y
jE4LTkxZmUtNTNkZDEwOWQ3NGY1IiwidXBuIjoiZnJhbmttQGNvbnRvc28uY29tIiwidW5pcX
VlX25hbWUiOiJmcmFua21AY29udG9zby5jb20iLCJzdWIiOiJKV3ZZZENXUGhobHBTMVpzZjd
5WVV4U2hVd3RVbTV5elBtd18talgzZkhZIiwiZmFtaWx5X25hbWUiOiJNaWxsZXIiLCJnaXZl
bl9uYW1lIjoiRnJhbmsifQ."
}
```

Imagen 10.- Token de autorización.

Para poder hacer las llamadas a la Office 365 Management Activity API únicamente hará falta añadir este token en la llamada con el formato Authorization: Bearer <token>.

Puedes encontrar un ejemplo de utilización de la API en el siguiente repositorio de GitHub: https://github.com/fchopo/BigBrotherOffice365.

> ***"los tipos de contenido y las operaciones existentes podemos extraer valiosa información que nos puede servir para realizar informes"***

Conclusión

En este artículo hemos visto cómo es posible consultar el registro de actividad de cualquier tenant de Office 365 mediante la Management Activity API. A partir de los tipos de contenido y las operaciones existentes podemos extraer valiosa información que nos puede servir para realizar informes o llevar a cabo auditorias sin los límites de tiempo que nos implica la consulta desde la interfaz gráfica del Office 365 Security & Compliance o el comando PowerShell Search_UnifiedAuditLog.

Como posibles mejoras propondríamos la incorporación de nuevos tipos de contenido, e incluso un período más extenso para la retención de los Content BLOBs.

FERRAN CHOPO GARCIA
IT Consultant & Trainer
ferran@ferranchopo.com
@fchopo
http://www.ferranchopo.com

46

Enable custom script on modern SharePoint sites to recover your favourite functionalities

Let me start with a HUGE disclaimer: even though it is possible to enable custom scripts on modern SharePoint, you should only consider using it when there is no other alternative.

Having custom script disabled by default on SharePoint removes functionalities that old school users miss, such as list templates and the solution gallery, among several others. Most of the features removed by custom scripts can be achieved using PnP and SharePoint Online Cmd-lets.

"even though it is possible to enable custom scripts on modern SharePoint"

If you need to bring back any of the functionalities listed below you will need to follow the steps described in the article below:

1.– Enable custom scripts on tenant level

2.– Enable custom script on SharePoint sites

SITE FEATURE	BEHAVIOUR	NOTES
Save Site as Template	No longer available in Site Settings	Users can still build sites from templates created before custom script was blocked.
Save document library as template	No longer available in Library Settings	Users can still build document libraries from templates created before custom script was blocked.
Solution Gallery	No longer available in Site Settings	Users can still use solutions created before custom script was blocked.
Theme Gallery	No longer available in Site Settings	Users can still use themes created before custom script was blocked.
Help Settings	No longer available in Site Settings	Users can still access help file collections available before custom script was blocked.
HTML Field Security	No longer available in Library Settings	Users can still use HTML field security that they set up before custom script was blocked.
Sandbox solutions	Solution Gallery is no longer available in Site Settings	Users can't add, manage, or upgrade sandbox solutions. They can still run sandbox solutions that were deployed before custom script was blocked.
SharePoint Designer	Pages that are not HTML can no longer be updated. Handling List: **Create Formand Custom Action** will no longer work. Subsites: **New Subsite** and **Delete Site** redirect to the **Site Settings** page in the browser. Data Sources: **Properties**button is no longer available.	Users can still open data sources.
Uploading files that potentially include script	The following file types can no longer be uploaded to a library .asmx .ascx .aspx .htc .jar .master .swf .xap .xsf	Existing files in the library are not impacted.

Source - https://docs.microsoft.com/en-gb/sharepoint/allow-or-prevent-custom-script?redirectSource-Path=%252fen-us%252farticle%252fAllow-or-prevent-custom-script-1F2C515F-5D7E-448A-9FD7-835DA935584F

Enable custom scripts on tenant level

Before getting custom script running on your modern site, you need to make sure it is already enabled at tenant level. To verify or enable it, you should follow these steps:

1.– From your SharePoint site, click on the waffle icon

2.– Click on Admin

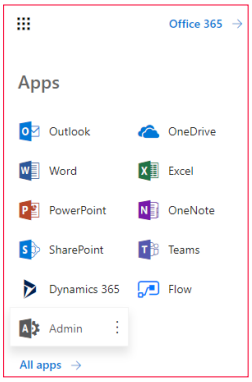

Image 1.- Accessing the Office 365 Admin Center from the App Launcher

3.– Expand Admin centers and click on SharePoint

4.– On the SharePoint Admin Center click on Settings

5.– Scroll down and locate the Custom Script

6.– Click on Allow users to run custom script on personal sites

7.– Click on Allow users to run custom script on self-service created sites

Image 2.- Enabling custom script in the SPO Admin Center

8.– Click OK

Note: Changes to the tenant settings can take up to 24 hours to be applied.

"Having custom script disabled by default on SharePoint removes functionalities that old school users miss"

Enable custom script on Share-Point sites

To enable custom scripts on modern sites, you will need to use SharePoint Online Cmdlets. If you don't have it installed download it from here.

https://www.microsoft.com/en-us/download/details.aspx-?id=35588

Once installed:

1.– Open the PowerShell console

2.– Execute the command

```
Connect-SPOService -Url https://tenant-admin.sharepoint.com
```

3.– Execute the command bellow with the url for the modern site collection

```
Set-SPOSite -Identity https://tenant.sharepoint.com/sites/contoso -DenyAddAndCustomizePages 0
```

Conclusion

Don't enable custom scripts unless it's your last option to recover a feature, having custom scripts enabled can put your tenant at risk.

JOAO FERREIRA
SharePoint Team Lead @ BindTuning
Twitter: @joao12ferreira
Blog: http://handsonsharepoint.net

45

Nosotros

Alberto Diaz

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

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. Pertenece 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 distintitas 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.siderys.com
Email: fabiani@siderys.com.uy
Blogs: http://blog.siderys.com
Twitter: @fabianimaz

Gustavo Velez

Gustavo Velez es Ingeniero Mecánico y Electrónico; trabaja en la arquitectura, diseño e implementación de sistemas de IT basados en tecnologías de Microsoft, especialmente SharePoint, Office 365 y Azure.

Propietario del sitio especializado en información sobre SharePoint en español http://www. gavd.net, autor de ocho libros sobre SharePoint y sus tecnologías y numerosos articulos y conferencias sobre el tema.

Sitio Web: http://www.gavd.net
Email: gustavo@gavd.net
Blogs: http://geeks.ms/blogs/gvelez/

Juan Carlos González Martín

Ingeniero de Telecomunicaciones por la Universidad de Valladolid y Diplomado en Ciencias Empresariales por la Universidad Oberta de Catalunya (UOC). Cuenta con más de 14 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 & Office 365. Juan Carlos es MVP de Office Apps & Services y co-fundador del Grupo de Usuarios de SharePoint de España (SUGES, www.suges.es), delGrupo de Usuarios de Cloud Computing de España (CLOUDES) y de la Comunidad de Office 365. Hasta la fecha, ha publicado 11 libros sobre SharePoint & Office 365, así como varios artículos encastellano y en inglés sobre ambas plataformas.

Email: jcgonzalezmartin1978@hotmail.com
Blogs: http://geeks.ms/blogs/jcgonzalez &
http://jcgonzalezmartin.wordpress.com/

Santiago Porras

Innovation Team Leader en ENCAMINA, lidera el desarrollo de productos mediante tecnologías Microsoft. Se declara un apasionado de la tecnología, destacando el desarrollo para dispositivos móviles y web, donde ya cuenta con 16 años de experiencia.

Microsoft MVP in Developer Technologies, colabora con las comunidades de desarrolladores desde su blog personal **http://geeks.ms/santypr** y ocasionalmente en **CompartiMOSS.com**. Además, es uno de los coordinadores de TenerifeDev, grupo de usuarios de .NET en Tenerife (**http://www.tenerifedev.com**)

Sitio Web: http://www.santiagoporras.es
Email: santiagoporras@outlook.com
Blogs: http://geeks.ms/santypr
Twitter: @saintwukong

Coordinadores de sección

GASTÓN CRUZ
Coordinador de PowerBi
gastoncruz@gmail.com

XAVIER MORERA
Coordinador de .Net
xavier@familiamorera.com

PABLO PERALTA
Coordinador de Dynamics CRM
pablop2006@gmail.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 tecnologías de Microsoft en español, todo el contenido deberá ser directamente relacionado con Microsoft y escrito en castellano. No hay limitaciones sobre el tipo de articulo o contenido, lo mismo que sobre el tipo de tecnología.
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: Artículos cortos: Máximo 1000 palabras y 1 o 2 figuras. Describen rápidamente una aplicación especial de alguna tecnología de Microsoft, o explica algún punto poco conocido o tratado. Experiencias de aplicación 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 tecnologías de Microsoft, 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) 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


Comparti
MOSS