DIGITALIZACIÓN DE PROCESOS CON HERRAMIENTAS DE OFFICE 365

Guillermo Rapado Gallego

Tutor: Roberto García Fernández

Máster en Ingeniería de Telecomunicaciones

Escuela Politécnica de Gijón

2018-2020

ÍNDICE

1.- INTRODUCCION 1
2.- JUSTIFICACIÓN 2
3.- INDUSTRIA 4.0 6
3.1.- IoT Internet Of Things 7
3.1.1.- Capa Física 8
3.1.2.- Capa de Conexión 9
3.1.3.- Capa Digital 9
3.2.- Big Data Analysis 10
3.2.1.- Sistemas de Análisis Existentes 11
3.2.2.- Relación entre IoT y Big Data Análisis 12
4.- HERRAMIENTAS 14
4.1.- Excel 14
4.1.1.- VBA 14
4.1.2.- Power Query 15
4.2.- Power Bi 15
4.2.1.- Obtención de datos 16
4.2.2.- Modelación de datos 18
4.2.3.- Actualización de datos 18
4.2.4.- Exportación de datos 21
4.3.- Power Automate 23
4.4.- Powershell 24
5.- ARQUITECTURA INICIAL 25
5.1.- Bases de datos. 25
5.1.1.- SQL database 26
5.1.2.- NoSQL database 27
5.2.- Localización de la base de datos 29
6.- AUTOMATIZACIÓN 32
6.1.- Análisis Bi de datos 34
6.2.- Generación de informes periódicos 39
6.3.- Tratamiento de correos 46
6.3.1.- Detección de posibles coincidencias 46

6.3.2.- Comparación con la base de datos 49
6.3.3.- Archivos adjuntos 51
6.4.- Presupuesto aproximado 52
7.- PROBLEMAS 54
8.- MEJORAS 59
9.- CONCLUSIONES 62
10.- BIBLIOGRAFIA 63

FIGURAS

Figura 2.1.- Protocolos POP e IMAP. 3
Figura 3.1.-Evolución de la industria. 6
Figura 3.2- Capas de IoT. 8
Figura 3.3- Relación entre IoT y Big Data Análisis. 13
Figura 4.1- Macro adjuntada a un botón. 15
Figura 4.2- Ejemplo de informe de Power Bi. 16
Figura 4.3- Obtención de datos con Power Bi. 17
Figura 4.4- Pasos comunes de Power Query. 18
Figura 4.5- Explicación de Microsoft de la puerta de enlace. 18
Figura 4.6- Extensión de Power Bi en Excel. 21
Figura 4.7- Tabla dinámica en Excel. 22
Figura 4.8- Cubo OLAP. 22
Figura 4.9- Ejemplo de flujo. 23
Figura 5.1- . SQL vs NoSQL. 25
Figura 5.2- Estructura de una base de datos SQL. 27
Figura 5.3- Crecimiento horizontal vs vertical. 28
Figura 5.4- Comparación superficial de diversos montajes de automatización. 30
Figura 5.5- Situación del ODBC. 31
Figura 6.1- Montaje final del proyecto. 33
Figura 6.2- Relaciones entre tablas en Power Bi. 35
Figura 6.3- Tabla usando las relaciones. 35
Figura 6.4- Objeto filtros de Power Bi. 36
Figura 6.5- DrillThough de Power Bi. 37
Figura 6.6- Informe con (derecha) y sin (izquierda) aplicarse el drillthough. 37
Figura 6.7- Informe sin (izquierda) y con (derecha) filtro. 39
Figura 6.8- Ejemplo aplicación del código de tablas dinámicas 41
Figura 6.9- Programador de tareas de Powershell. 45
Figura 6.10- Bloque 1 del flujo de correos. 47
Figura 6.11- Bloque 2 del flujo de correos. 49

Figura 6.12- Bucle de comprobación de coincidencias. 50
Figura 6.13- Interior del bloque condicional del asunto. 50
Figura 6.14- Interior del bloque condicional del cuerpo. 51
Figura 6.15- Bloque 3 del flujo de correos. 52
Figura 7.1- Error porque el Mashup no termino la conexión con el Excel (arriba) y Excel intentando guardarlo con otro nombre (abajo). 55
Figura 7.2- Método de envío directo explicado por Microsoft. 57
Figura 7.3- Solución final para los correos diarios. 58
Figura 8.1- Análisis de Microsoft de Power Bi Embedded. 60
Figura 8.2- Imagen de ejemplo de Microsoft de Power Bi Embedded. 61

TABLAS

Tabla 2.1-Comparación de licencias de Microsoft 365. 4
Tabla 4.1- Lista de FQDNs usados por la puerta de enlace. 20
Tabla 6.1 – Tabla de precio estimado mínimo. 53

1.- INTRODUCCION

En el presente trabajo titulado Digitalización de procesos con herramientas de Office 365 se responde a la necesidad de muchas empresas de pequeño o mediano tamaño al deseo de pegar el salto a la llamada industria 4.0. Para muchas empresas normalmente no es posible reservar una gran cantidad de capital para invertir en digitalizarse, pero siguen teniendo que, por ejemplo, generar los informes diarios de ventas de una empresa, donde alguien tiene que reservar todos los días las primeras horas de la mañana para realizar un informe del día anterior.

En este proyecto se persigue mostrar formas sencillas de automatizar algunos de los procesos que toda empresa realiza con reiteración aprovechándose de las herramientas proporcionadas por Microsoft. Actualmente muchas empresas están optando por contratar los servicios de Microsoft 365 para empresas ya que te permite usar las herramientas de Office, además de Outlook y OneDrive desde la web. Sin embargo, eso no es todo lo que viene en los paquetes de Microsoft 365 y, combinándolo con lo que ya se manejaba de Office, se abre la puerta a una gran posibilidad de automatizaciones.

Así que en las siguientes páginas se mostrarán varios ejemplos de procesos automatizados utilizando en todas las situaciones posibles programas de Microsoft. Si se usase cualquier otro tipo de licencias de Microsoft, ya que tienen alguna utilidad no incluida en la licencia de Microsoft 365, se comentará junto con las ventajas respecto a otra opción que no la necesitase.

2.- JUSTIFICACIÓN

En este trabajo se presentan las experiencias que se han obtenido trabajando en una empresa que se dedica a proporcionar asistencia tecnológica a empresas que no pueden permitirse tener un informático propio, o no necesita uno a tiempo completo, o que quieren tener un proveedor con el que tener una relación cercana y de confianza. A medida que pasaba el tiempo cada vez más empresas contrataban migraciones de sus dominios de los servicios de correos a Microsoft, esto se debe a que normalmente estas empresas contratan otras empresas especializadas que, como Microsoft, te proporcionan un dominio y espacio en sus servidores para almacenar todos los buzones de correo que crees con ese dominio. El cambio de debía a que, con estas empresas se utilizan generalmente unos tipos de protocolos llamados POP o IMAP, mientras que Microsoft utiliza Exchange con su oferta de Microsoft 365:

- El protocolo POP (Post Office Protocol), cuya versión más actual es POP3, crea una copia local del correo que lees y lo borra del servidor donde esté almacenado el correo original. Pero esto es tanto su punto fuerte como su desventaja, ya que al eliminar el correo una vez descargado el buzón del correo en el servidor que se tenga contratado no se llenará y no será necesario contratar más espacio. Sin embargo, los correos solo podrán verse en el ordenador donde se hayan descargado y si el archivo se corrompiese o se perdiese también se perderían los correos.
- IMAP (Internet Message Access Protocol) a diferencia de POP sincroniza las carpetas de correo del servidor con varios dispositivos, lo que permite acceder a los correos desde varios sitios, como se ve en la figura 2.1, y que si se lee desde uno queda marcado como leído en el resto. Pero al no eliminarse los correos el espacio asignado para el buzón este se llenará, necesitándose contratar más espacio o eliminar los correos a mano.
- Y por último Exchange que además de los correos, como IMAP, sincroniza también los contactos y calendarios.

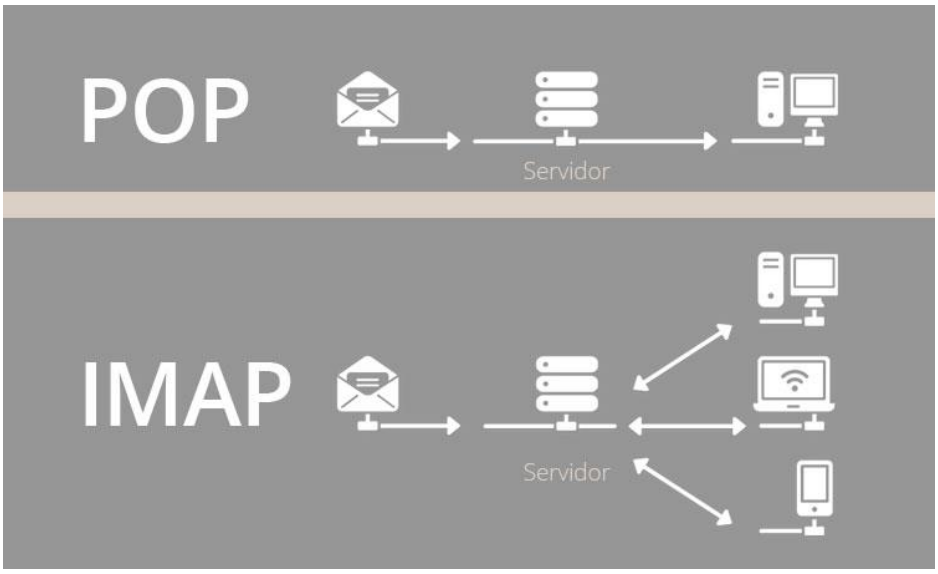

Figura 2.1.- Protocolos POP e IMAP.

Se podría entrar más en detalle, pero lo importante es saber que normalmente en estos tipos de servicios se configura el correo con el protocolo POP ya que los precios de los buzones sueles ser muy altos, y en una empresa que vive de los correos con los clientes. Pero el hecho de solo poder ver los correos desde el dispositivo que tenga el archivo con los correos, como .pst (los archivos creados por Microsoft Office para almacenar las carpetas de los buzones configurados), no solo limita la libertad de trabajo si no que, para evitar la pérdida de los correos, se suelen realizan copias de seguridad de dichos archivos, trasladando los problemas de almacenamiento al entorno local. Por eso, para estos tipos de empresa, los protocolos IMAP y Exchange arreglan muchos de los problemas que se tenía, pero el aumento de coste exponencial por el almacenamiento de los buzones de cada empleado que no utilice POP es una de las razones por las que a muchas empresas les cuesta dar este primer paso.

Y para empresas en estas situaciones es donde entra Microsoft con sus ofertas de Office 365, que te permite, con una sola licencia, tener un buzón con gran espacio y crear varios buzones compartidos con espacios de almacenamiento más reducido (contratando tarifas solo para Outlook). Además, Microsoft te facilitan la migración del dominio del servicio en el que estabas al suyo.

Otra razón por la que Microsoft 365 es atractivo para las empresas es que incluye licencias de Office, ya que Word y Excel son herramientas muy usadas en el campo administrativo. Pero eso no es todo lo que viene incluido con las licencias de Microsoft 365, si no que vienen otras herramientas menos conocidas de Microsoft, pero con mucho potencial para facilitar la vida de muchos trabajadores.

Para el análisis de precios de la Tabla 2.1 y de las distintas licencias se van a tener en cuenta únicamente las de Microsoft 365 Empresa:

	Microsoft 365 para Empresas	Microsoft 365 E1	Microsoft 365 E3	Microsoft 365 E5
€/usuario/mes	12,80 €	6,70 €	19,70 €	34,40 €
Office	Incluido	Solo versión web	Incluido	Incluido
Correo	Solo Outlook	Exchange con 50 GB de buzón	Exchange con 100 GB de buzón	Exchange con 100 GB de buzón
Onedrive	1 TB	1 TB	1-5 TB	1-5 TB
Teams	Incluido	Incluido con todas las extensiones	Incluido con todas las extensiones	Incluido con todas las extensiones
Sharepoint	No incluido	Plan 1	Plan 2	Plan 2
Administrador de tareas	-Microsoft Forms	-Power Apps -Power Automate -Microsoft Form	-Power Apps -Power Automate -Microsoft Form	-Power Apps -Power Automate -Microsoft Form
Power bi Pro	No incluido	No incluido	No incluido	Incluido

Tabla 2.1-Comparación de licencias de Microsoft 365.

Esta tabla contiene las características más significativas de las licencias para este proyecto, pero no representa todo lo que ofrece cada una de ellas.

Para muchas empresas este primer paso es suficiente: buzones de correo sincronizados, licencias de Office y gran espacio de almacenamiento en OneDrive. Sin embargo, esto es solo el comienzo del avance a una empresa más tecnológicamente organizada: calendarios compartidos por Outlook, separar la empresa en grupos con Sharepoint, mandar encuestas a los clientes con Forms, etc. Utilizando las herramientas ya creadas y orientadas a uso general de Microsoft (como las comentadas en los ejemplos) se puede facilitar el trabajo del día a día a los trabajadores.

Pero estos avances están sostenidos por herramientas de Microsoft ya diseñadas para ser sencillos y no requerir muchos conocimientos técnicos para montajes más complejos, para realizar montajes más complejos y personalizados se requiere usar el resto de las herramientas proporcionadas por las licencias de Microsoft.

Y ahí es donde ayuda la empresa, una vez vendida la licencia de Microsoft 365 a las empresas se les ofrece realizar un montaje personalizado que automatice una función

importante pero repetitiva que realizase en su empresa, aprovechando las herramientas que ya han pagado pero que no tenían intención de usar. Así se le da más utilidad por el mismo precio a las licencias de Microsoft y se ayuda a la empresa a realizar un pequeño primer salto hacia una empresa más digital que tanto les cuesta dar a empresas de pequeño o mediano tamaño pero que ya están muy bien establecidas.

Tras realizar varios proyectos de este ámbito para distintas empresas se pudo comparar la eficiencia de varios métodos de automatización usando las herramientas a nuestra disposición, recogiendo dicha experiencia en este trabajo, junto a los problemas que surgieron de usar dichas herramientas.

3.- INDUSTRIA 4.0

Los valores que surgieron durante los primeros momentos de la industrialización están siendo moldeados por los desarrollos surgidos durante la cuarta etapa de la industrialización, llamada Industria 4.0, representada la evolución hacia ella en la figura 3.1. Una continuación de la tercera revolución industrial, la cual comenzó a principios de 1970 gracias a los avances en electrónica y la tecnología que permitieron realizar un gran nivel de automatización en la fabricación en las empresas. En 2014 Alemania encabezó el avance hacia esta cuarta revolución industrial con su innovación de los servicios y en Cyber-Physical System-enables manufacturing [1].

El desarrollo hacia la industria 4.0 está teniendo una influencia sustancial en las industrias de manufacturación, basándose en el establecimiento de fábricas, productos y servicios inteligentes integrados en el llamado Internet de las Cosas (IoT) y de los servicios llamados Industrial Internet. Este desarrollo hacia la industria 4.0 presenta grandes oportunidades para el desarrollo de una fabricación sostenible haciendo uso de la gran cantidad de información producida y obtenida y de la Infraestructura Común de Telecomunicaciones (ICT) [2].

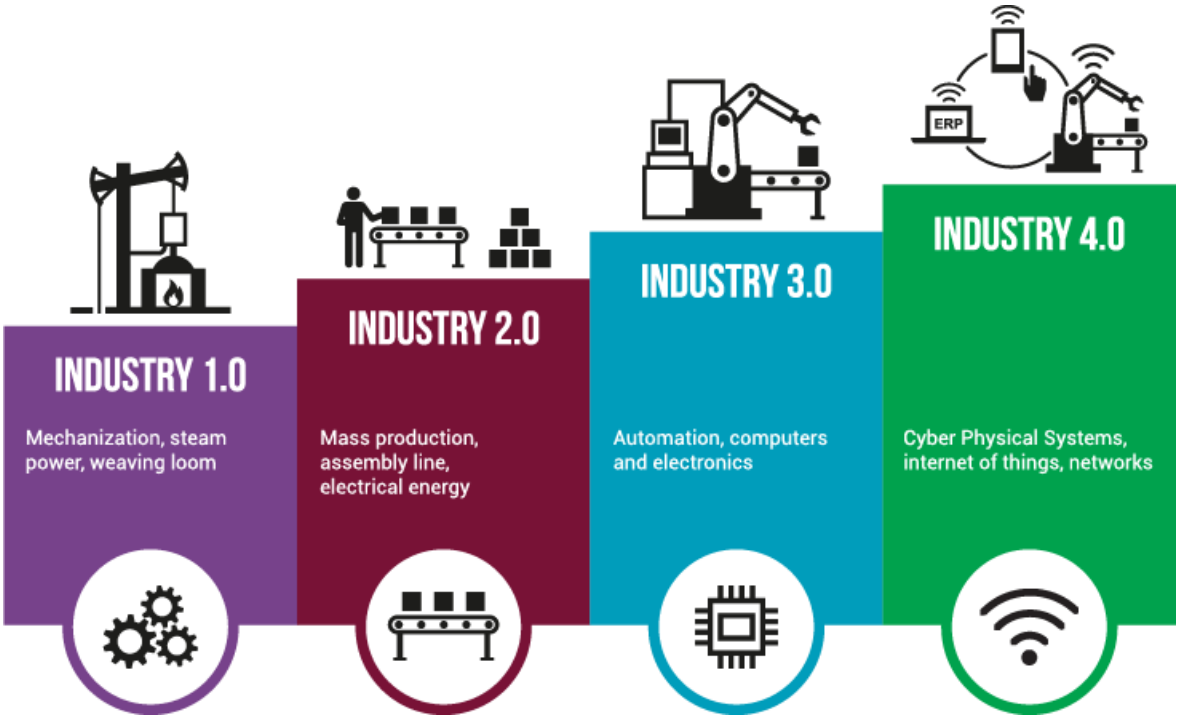

Figura 3.1.-Evolución de la industria.

De las tecnologías existentes de la industria digital abarcadas por la industria 4.0 están:

- El Internet de las Cosas (IoT).
- La Robótica autónoma.
- Computación en nube.
- Big Data Analysis.
- Fabricación aditiva.
- Integración de sistemas.
- Realidad aumentada.
- Simulaciones.
- Seguridad cibernética.

Esta fusión de las Tecnologías de la Información (TI) y de las Tecnologías de la Operación (OT), que hasta hace poco eran tradicionalmente independientes, ofrece un gran potencial de automatización mejorando de forma masiva los procesos y la optimización de la producción[3]. Esta innovación resultante está conduciendo al desarrollo de nuevos modelos comerciales y a nuevas fuentes de ingresos, con fundamento en la información y los servicios prestados.

Para la mayoría de los fabricantes esta industria 4.0 no es más que una visión de futuro, sin embargo, la transformación digital ya ha comenzado gracias a los grandes avances en robótica y a la impresión 3D, que han producido un gran impacto en el mundo de los metales y los plásticos. Ya se están obteniendo beneficios y mejoras en cuanto a la flexibilidad, calidad, seguridad y velocidad de producción en varios campos. Sin embargo, con estos avances han surgido nuevos desafíos, especialmente en el mundo de la información, programación de producción y seguridad cibernética.

3.1.- IoT Internet Of Things

El Internet de las Cosas (IoT) es un componente importante para las empresas de hoy en día para la automatización y la conexión de varios componentes dentro de una economía. En resumen, IoT es la interconexión de red entre dispositivos físicos: vehículos, edificios u otros elementos que vengan integrados con electrónica, sensores, software y actuadores que les den conectividad y les permitan recopilar y compartir información entre ellos.

Este concepto gira en torno a redes de sensores que recopilan información de recursos muy distintos, desde relojes hasta humidificadores, y tratar toda la información obtenida en local o la nube, dependiendo del modelo de negocio, y así proporcionar un valor distinto al producto tanto para el consumidor como para la corporativa [4].

En la Figura 3.2 se muestra un ecosistema básico de IoT donde se diferencian sus distintas capas. En cada una de las capas se mezcla hardware, software y parte del servicio proporcionado para crear una cadena entre cada una de ellas.

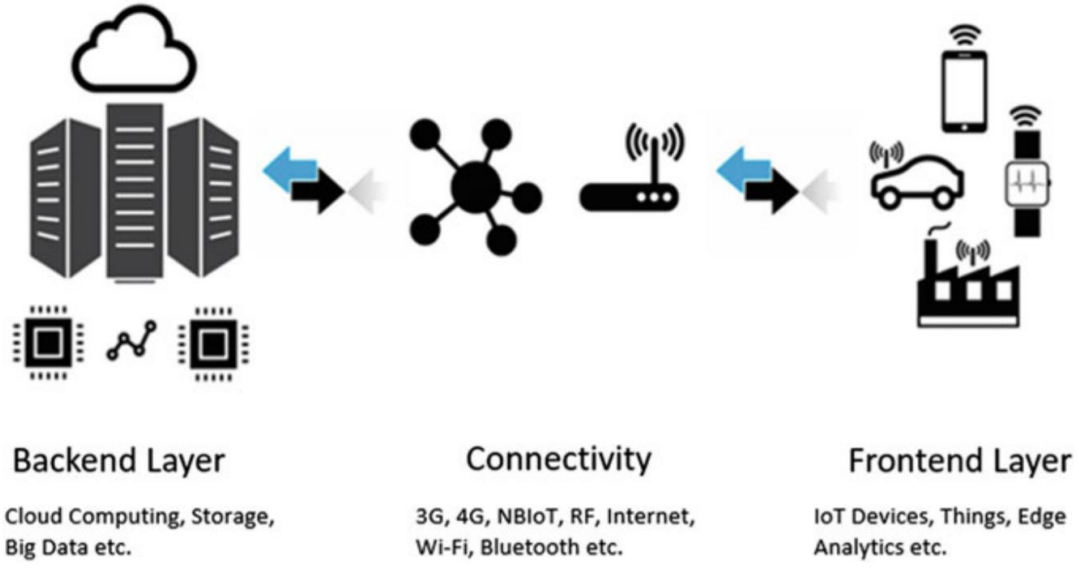

Figura 3.2- Capas de IoT.

Gracias a los avances y proliferación de los sensores se ha creado un mundo cada vez más cuantificado y accesible, los datos obtenidos de forma automática de esta forma permiten, a los dispositivos interconectados, transformar el mundo físico en uno digital. Mediante el análisis de esta gran cantidad de información, obtenida del mundo digital creado por los sensores, es posible entender las tendencias actuales del mercado o detectar cuando va a producirse un problema. Así, las empresas o los países pueden producir proyecciones de mercado y reaccionar más rápido frente a problemas. Esto explica brevemente como IoT crea un gran valor para las empresas y la economía en general [5].

Como ya se comentó en la figura anterior, se presentan las distintas capas de un ecosistema IoT, que se dividen en las capas: física, de conexión y digital.

3.1.1.- Capa Física

En la capa física es donde están situados los sensores y los microcontroladores. Estos trabajan juntos para proporcionar uno de los aspectos más importantes del Internet de las Cosas (IoT): detectar los cambios en el objeto en el entorno donde están situados para así capturar los datos más relevantes en tiempo real.

Los sensores se utilizan para detectar cambios físicos, como de temperatura, humedad, presión, luz o movimiento, además de para detectar la relación de un objeto con otro y en del medio ambiente en si, como la presencia o ausencia de un objeto. En resumen, la ubicación de un objeto rastreable o cualquier tipo de información que pueda ser útil para la empresa que lo organizo.

Los actualizadores son otros dispositivos críticos de la capa física que interactúan con el entorno o un objeto, como por ejemplo el controlador de luces en una carretera que se tiene que encender cuando pasa una persona.

Las redes de malla están formadas por una serie de repetidores fáciles de instalar ubicados en lugares elevados, como farolas o postes telefónicos, para formar una cama sobre la zona de la que se desea obtener información.

3.1.2.- Capa de Conexión

La capa de conexión es la responsable de conectar entre ellas objetos inteligentes, como dispositivos de red y servidores. Además, también se encarga de transmitir y procesar los datos de los sensores.

Los dispositivos IoT se conectan y comunican usando diversas modelos y tecnologías, como redes IP, Bluetooth, 3G, 4G, Z-Wave, WIFI, RFID o NFC (del Internet Society IoT Inform 2015) [6].

- El modelo de comunicación de dispositivo a dispositivo presenta 2 o más dispositivos que se comunican directamente entre sí, sin un servidor de aplicaciones intermedio.
- El modelo de comunicaciones de dispositivos a nube consiste en un dispositivo que se conecta directamente a un servicio en la nube de Internet, como con un proveedor de servicios de aplicaciones, para intercambiar datos y poder controlar el tráfico de mensajes.
- El modelo de dispositivo a puerta de enlace o modelo de enlace de dispositivo a capa de aplicación (ALG) conecta a través de un servicio ALG, usándolo como conductor, al servicio en la nube.
- El modelo de intercambio de datos de fondo usa una arquitectura de comunicación para exportar y analizar los datos desde un servicio de nube, combinándolos con los datos de otras fuentes.

3.1.3.- Capa Digital

La capa digital es la encargada de almacenar, analizar y procesar grandes cantidades de datos que llegan desde la capa de conexión. En ella se puede administrar y proporcionar un conjunto de servicios diverso para las capas inferiores empleando tecnologías como bases de datos, computación en la nube y módulos de proceso de Big Data.

Esta capa también es responsable de entregar la aplicación que proporciona el servicio específico a los usuarios y es donde residen varias aplicaciones en las que IoT puede ser desplegado, como, por ejemplo: casas inteligentes, salud o ciudades inteligentes.

Es decir, en esta capa es donde se produce el análisis de los datos obtenidos de los sensores o de las fuentes de información de la capa física, donde se produce la automatización y donde se entrega el producto final al usuario. Por eso, es en esta capa donde

se trabajó en este proyecto para realizar las automatizaciones a través de la información almacenada en las bases de datos de los clientes.

3.2.- Big Data Analysis

Como se comentó, la gran cantidad y variedad de datos asociados con el Internet de las Cosas (IoT) surgió gracias a la extensión de los sensores y las diferentes fuentes de datos. Esta variedad también aumento la complejidad de los datos e información obtenidos y del modelado que hay detrás de ellos.

Esta variedad en los datos creo nuevos requisitos de almacenamiento para los propios datos y para sus bases de datos, donde el formato de los datos debe adaptarse dinámicamente. Además, los datos deben ser protegidos durante su ciclo de vida, desde su recolección en fuentes confiables hasta su procesamiento en instalaciones de computo confiables.

La veracidad de la información en el Internet de las Cosas (IoT) también mejoró a medida que aumento la calidad de los sensores y de los datos con el tiempo. Por ejemplo, el uso de sistemas de almacenamiento y recuperación de datos, como el RFID (Identificación por Radiofrecuencia), genera información mucho más confiable que el de hace una década [7].

Tales volúmenes de datos, junto su velocidad de crecimiento y de variedad, genera una gran cantidad de información que requieren un procesado analítico. Situaciones donde sistemas de almacenamiento de bases de datos tradicionales no serían capaces de procesar y analizar esta, rápidamente creciente, cantidad de información.

El Análisis de Big Data es el proceso de examinar un gran conjunto de datos y mostrar patrones invisibles, como correlaciones ocultas, tendencias de mercado, preferencias de los clientes e información comercial útil [8]. La capacidad de poder analizar una gran cantidad de datos puede ayudar a una organización a manejar información que podría afectar en gran medida a su negocio, acelerando la detección de un estado deseado o detectando cuando es mejor vender un producto. Por lo tanto, el objetivo principal de Análisis de Big Data es ayudar a asociaciones empresariales a tener una mejor comprensión de sus datos, y que así, al tener una mejor comprensión y base de la situación en la que se encuentran, puedan tomar decisiones empresariales más eficientes y bien informadas.

Además, el Análisis de Big Data permite a científicos e investigadores analizar una gran cantidad de datos que no podrían haber podido aprovechar con herramientas tradicionales.

Para ello, el Análisis de Big Data requiere unas herramientas y tecnologías que puedan transformar todos esos datos en formatos estructurados, no estructurados o

semiestructurados en un formato de datos y metadatos comprensibles para su analizado. Los algoritmos utilizados en estas herramientas deben ser capaces de descubrir patrones, tendencias y correlaciones en una gran variedad de datos en un horizonte temporal de datos. Después de analizar estos datos, estas herramientas deberán visualizar los resultados en tablas, gráficos y tablas dinámicas que permitan su estudio y la toma de las decisiones correctas a causa de ellas.

Por eso, el Análisis de Big Data es un desafío para muchas aplicaciones existentes debido a la gran complejidad de los datos y a la escalabilidad de los algoritmos que admitan dichos procesos.

3.2.1.- Sistemas de Análisis Existentes

- El análisis en tiempo real, generalmente, se realiza con los datos recopilados de los sensores. En dichos casos, los datos recibidos cambian constantemente y requieres técnicas que los analicen en vivo para obtener el análisis rápido y poder actuar en consecuencia en el menor tiempo posible. Dos arquitecturas de ejemplo para el análisis en tiempo real son: grupos de procesamiento en paralelo que utilizan las bases de datos tradicionales y plataformas informáticas basadas en memoria.
- El análisis fuera de línea es utilizado cuando no es necesaria una respuesta rápida. Por ejemplo, muchas empresas de Internet utilizan este tipo de análisis para ahorrar el coste de la conversión de formato de los datos. SCRIBE y Kafka son algunos ejemplos de arquitecturas para analizar datos fuera de línea.
- El análisis de nivel de memoria se aplica cuando la memoria del cluster es mayor que el volumen de datos. Generalmente requiere varias tecnologías de bases de datos internas para mejorar la eficiencia del análisis.
- Al contrario, el análisis de BI se usa cuando el volumen de datos el mayor que el nivel de memoria, donde, en estos casos, los datos se importan a un entorno de análisis BI. Además, BI puede ayudar a descubrir oportunidades comerciales estratégicas a partir de la avalancha de datos y permite una fácil interpretación de ellos. Gracias a esto, identificar nuevas oportunidades y poder implementar estrategias más rápido puede proporcionar una gran ventaja en el mercado.
- Y el análisis masivo se usa cuando el tamaño de los datos es mayor incluso que la capacidad total de análisis de la herramienta BI y de las bases de datos tradicionales. Este utiliza el sistema de archivos distribuidos de Hadoop, una estructura de software de código abierto para almacenar datos y ejecutar aplicaciones en clusters, para almacenar los datos y el MapReducte, un modelo de programación para dar soporte a la computación paralela, para su análisis.

3.2.2.- Relación entre IoT y Big Data Análisis

El análisis de Big Data en IoT requiere procesar una gran cantidad de datos en vivo y almacenar dichos datos en varias tecnologías de almacenamiento distintas a la vez [9]. Dado que una gran parte de los datos no estructurados se recogen directamente en herramientas habilitadas para la web, su implementación en Big Data necesita analizar los datos con gran rapidez para grandes consultas. Es necesario para que la organización pueda obtener la información que desean y así poder tomar las decisiones rápidamente y poder interactuar con otras personas o dispositivos en respuesta.

La interconexión de dispositivos de actuación y de detección permite la capacidad de compartir información a través de plataformas con una arquitectura unificada y así desarrollar una imagen operativa común para permitir aplicaciones innovadoras. Así, la necesidad de integrar Big Data con IoT es convincente, habiendo sido reconocidas ya en el campo de TI y de negocios. Aunque el Internet de las Cosas (IoT) está creciendo con mucha más rapidez que Big Data Analysis actualmente, estas dos tecnologías son interdependientes y tienen que desarrollarse conjuntamente.

En general, como se comentó, IoT aumenta la cantidad de datos en cantidad y variedad, ofreciendo así la oportunidad para poder aplicar y desarrollar los análisis que maneja Big Data. Además, aplicar las tecnologías de Big Data en IoT acelera los avances de investigación y modelado comercial de IoT. Y, la relación entre Big Data e IoT, representada en la figura 3.3, se puede dividir en 3 partes para gestionar los datos de IoT.

- El primer paso comprende la gestión de fuentes de datos de IoT, donde los sensores usados utilizan aplicaciones para poder comunicarse entre ellos. Por ejemplo, almacenar de forma básica, con una aplicación de bajo coste, toda la información de tráfico, de usar objetos como cámaras CCTV o semáforos que proporcionan información en distintos formatos.
- El segundo paso es la generación de datos Big Data basadas en su volumen, velocidad y variedad. Estas grandes cantidades de datos se almacenarían en archivos Big Data en bases de datos compartidas que toleran fallos distribuidos.
- El último paso sería aplicar herramientas de análisis Query capaces de analizar grandes cantidades de datos de IoT con rapidez.

Figura 3.3- Relación entre IoT y Big Data Análisis.

4.- HERRAMIENTAS

En este apartado se explicarán las herramientas que se usaron para este trabajo y una breve explicación de su funcionalidad.

4.1.- Excel

Excel es, junto a Word, la herramienta más conocida de Microsoft. Es un software de hojas de cálculo que te da una gran libertad a la hora de trabajar con datos y con las fórmulas ya proporcionadas por Microsoft para usar. Además, Excel tiene una serie de aplicaciones que aumentan aún más las posibilidades de la herramienta:

4.1.1.- VBA

Microsoft VBA (Visual Basic for Applications) es un lenguaje de programación simple integrado en Excel, y otras herramientas de Office, cuya utilidad principal es la de crear macros que automaticen tareas sencillas. Esto lo haría perfecto para automatizar de forma sencilla tareas simples que tenga que realizar el trabajador de forma constante en una hoja de cálculo, como ajustar las dimensiones de una tabla automáticamente cada vez que se trabaja en ella.

Por ejemplo, el código siguiente te permite guarda la hoja de cálculo y cerrarla. Lo puedes poner en una macro en Excel y adjuntársela a diversos componentes, como el botón de la figura 4.1, para que al interactuar con él se ejecute la macro y realice la función deseada automáticamente. También se puede programar para ejecutarse al pulsar Ctrl más la tecla que decidas.

```
Private Sub CommandButton1_Click()

    ThisWorkbook.Save
    Application.Quit

End Sub
```

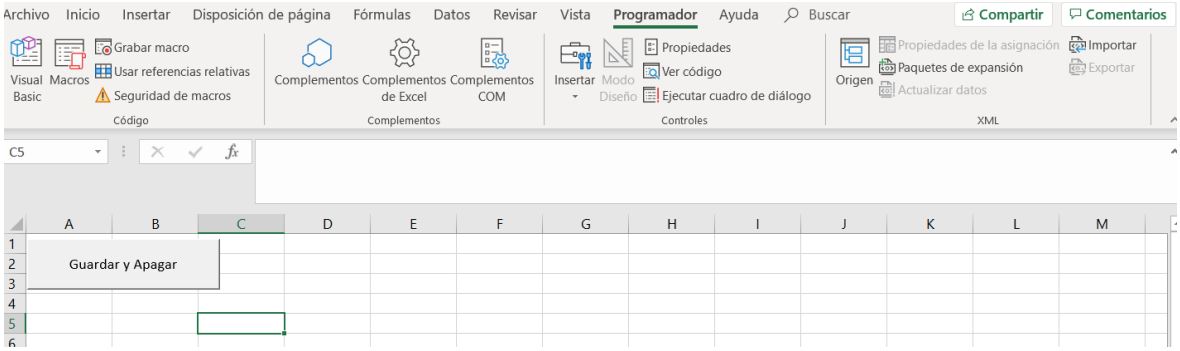

Figura 4.1- Macro adjuntada a un botón.

Es decir, VBA sirve para asistir a un trabajador que está usando Excel a realizar acciones recurrentes con rapidez. Por lo que, aunque existe la forma de crear una macro que se ejecute al arrancar la hoja de cálculo, requiere la existencia de una persona que interactúe con el programa para que funcione con libertad.

4.1.2.- Power Query

"Power Query es una tecnología de conexión de datos que permite detectar, conectar, combinar y refinar orígenes de datos para satisfacer sus necesidades de análisis. ", esta es la descripción que Microsoft da de Power Query. Con esta herramienta integrada en Excel es posible conectarse a nuestra base de datos, u otras Excel que queramos, para obtener los datos y poder manejarlos.

Power Query es una herramienta muy útil pues permite conectar con cualquier base de datos, incluso con servideros en el extranjero mientras tengamos conexión directa con ellos, filtrarlo como se desee. Además, dichas conexiones se pueden realizar en segundo plano, lo que convierte a esta tecnología en la más importante a la hora de automatizar cualquier proceso que incluya la búsqueda o el muestreo de datos.

Además, el hecho de que te permite relacionar datos y combinarlos de distintas tablas, incluso de distintos orígenes, le da ventaja a la hora de manejar la información de bases de datos de gran tamaño.

En cuanto al protocolo con el que se conecta a la base de datos, depende del tipo de conexión que se realice, SQL, ODATA, ODBC etc.

4.2.- Power Bi

Power Bi es una herramienta para analizar y presentar de forma interactiva grandes cantidades de datos a través de una interfaz gráfica web, como se muestra con un ejemplo en la figura 4.2:

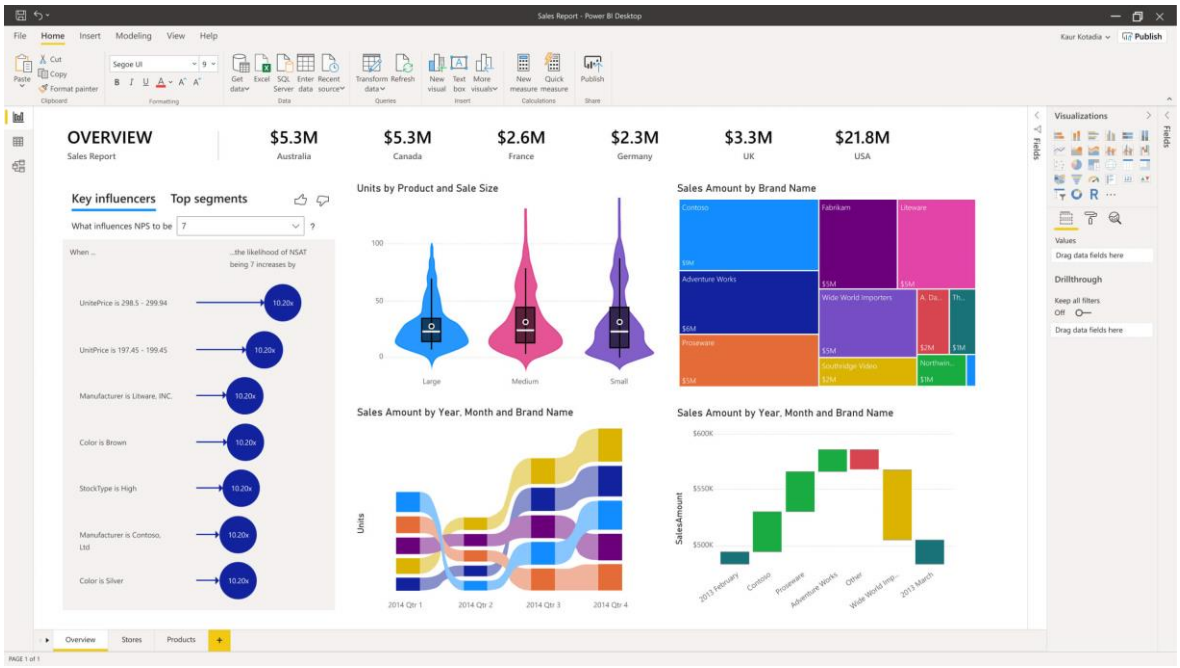

Figura 4.2- Ejemplo de informe de Power Bi.

Como herramienta es relativamente reciente, por lo que ha estado recibiendo muchos cambios recientes y se podría decir que aún está en desarrollo. Pero, a pesar de tener bastantes limitaciones, es capaz de manejar grandes cantidades de datos con rapidez.

Power Bi es una herramienta web, sin embargo, cuenta con una versión desktop que sirve como intermediario entre Power Bi web y la base de datos de donde se saca la información hasta su publicación definitiva (desde la web también es posible realizar la conexión, aunque con opciones más limitadas).

4.2.1.- Obtención de datos

Como con Excel, Power Bi tiene varias opciones para coger información dependiendo del origen de datos, como se puede ver en la figura 4.3:

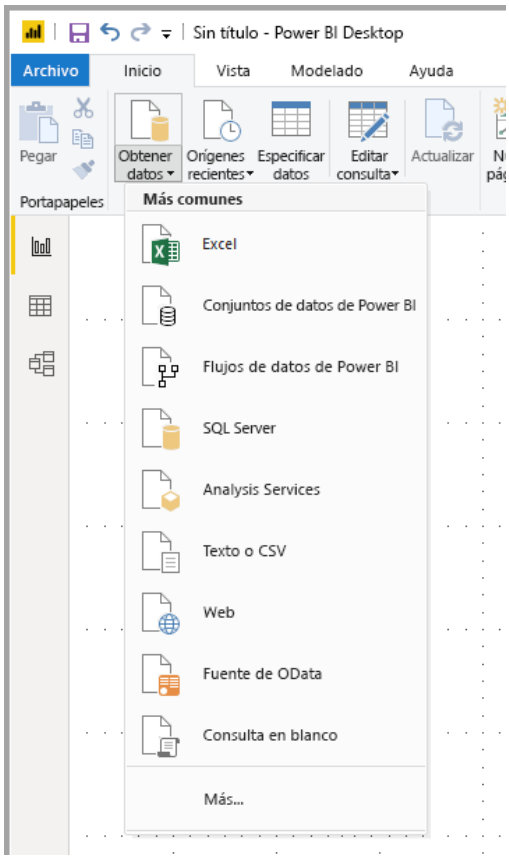

Figura 4.3- Obtención de datos con Power Bi.

Además, Power Bi te da la opción de elegir entre realizar la conexión con un Direct Query con la base de datos, importarla o con un Live Connection:

- Importarlo te permite manejar las relaciones entre tablas desde la versión de web. Como se descarga toda la información el tiempo de respuesta de los informes que se generen con ellos será más rápido, pero aumentara el espacio del modelo y si sobrepasa el límite de 1GB dejara de funcionar. Además, te da acceso al Query integrado de Power Bi para transformar los datos descargados, pero requiere volver a descargar los datos periódicamente.
- Con Direct Query Power Bi supone que ya se ha realizado toda la manipulación de información necesaria desde el origen y solo realiza consultas de la información deseada. Esto es útil porque así la información nunca sale de la base de datos, arreglando el problema de importar del límite de tamaño, pero aumentando el tiempo de respuesta. Esta opción solo se puede realizar con algunos tipos de orígenes de datos y no da acceso al Query de Power Bi.
- Y el ultimo es Live Connection, que solo funciona en 3 tipos de SSAS (SQL Server Analysis Services): Multidimensional, Azure Tabular y Tabular on premise. Este método de conexión es el más restrictivo ya que todo tipo de preparación de datos se hace desde el origen.

4.2.2.- Modelación de datos

Power Bi desktop, como Excel, funciona con Power Query, lo que te permite relacionar entre tablas, combinar, crear nuevas columnas, etc, como Microsoft muestra con la figura 4.4. Además, Power Bi cuenta con un mayor repertorio de fórmulas con las que trabajar, lo que expande las posibilidades.

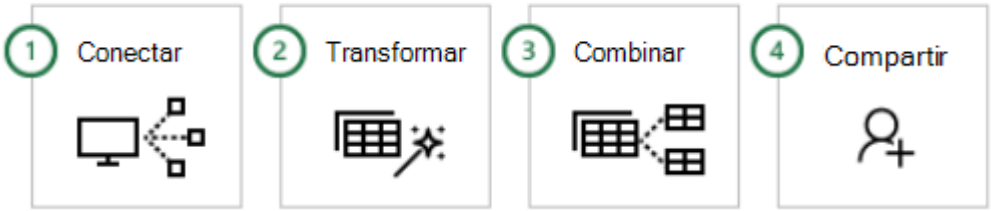

Figura 4.4- Pasos comunes de Power Query.

Una de las ventajas de Power Bi es que se pueden crear tablas dinámicas donde, con la ayuda de medidas que se puedes crear, se pueden hacer relaciones rápidas entre filas y columnas, ayudando así a una mejor representación de los datos.

4.2.3.- Actualización de datos

Power Bi, como ya se comentó, es una herramienta web, por lo que se necesita una forma de mantener la conexión con la base de datos, para eso Microsoft te proporciona una aplicación llamada puerta de enlace. Esta se instala en el servidor donde se encuentra el origen de los datos para así mantener una conexión entre esta y, como se representa en la figura 4.5, la herramienta. La puerta de enlace no es necesaria para las conexiones con otras herramientas web, como Sharepoint o Onedrive.

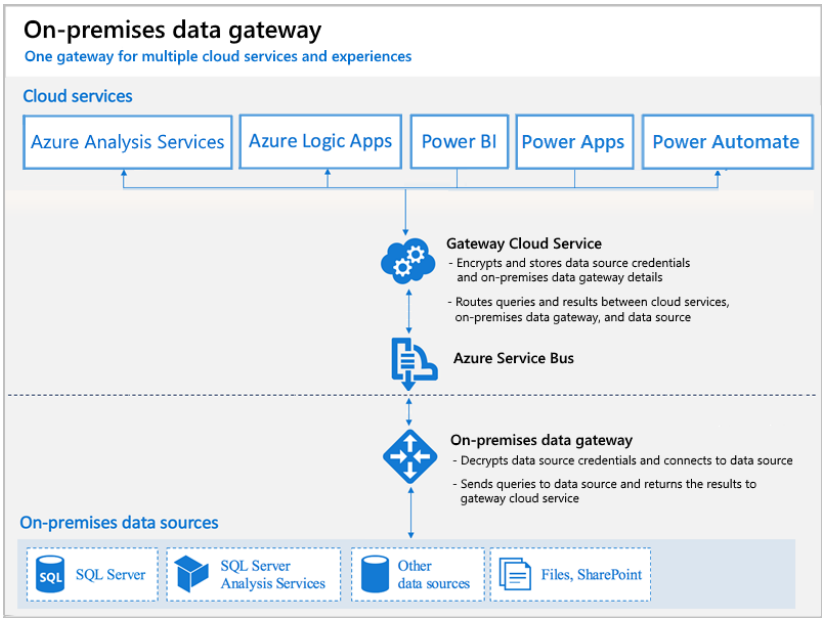

Figura 4.5- Explicación de Microsoft de la puerta de enlace.

La puerta de enlace queda funcionando en segundo plano en el servidor donde está instalado, esto significa que no requiere ningún tipo de interacción con el usuario una vez instalado.

La puerta de enlace se comunica con las herramientas usando una IP proporcionada por Microsoft dependiendo de la región, que se puede comprobar en su página web. Además, usa una serie de dominios FQDN (Fully Qualified Domain Name), representados en la tabla 4.1, para comunicarse por TCP con los servidores de Microsoft, o HTTPS si el usuario lo fuerza desde la aplicación o si le es imposible comunicarse por TCP.

Si se trabaja en un servidor es recomendable añadir los dominios en la lista de permitidos y abrir sus puertos para que no se produzcan errores en la comunicación que interfieran en la actualización de datos.

Dominios	Puerto usado	Descripción
*.download.microsoft.com	80	Used to download the installer. The gateway app also uses this domain to check the version and gateway region.
*.powerbi.com	443	Used to identify the relevant Power BI cluster.
*.analysis.windows.net	443	Used to identify the relevant Power BI cluster.
*.login.windows.net, login.live.com, and aadcdn.msauth.net	443	Used to authenticate the gateway app for Azure Active Directory (Azure AD) and OAuth2.
*.servicebus.windows.net	5671-5672	Used for Advanced Message Queuing Protocol (AMQP).
*.servicebus.windows.net	443 and 9350-9354	Listens on Service Bus Relay over TCP. Port 443 is required to get Azure Access Control tokens.
*.frontend.clouddatahub.net	443	Deprecated and not required. This domain will be removed from the public documentation as well.
*.core.windows.net	443	Used by dataflows to write data to Azure Data Lake.
login.microsoftonline.com	443	Used to authenticate the gateway app for Azure AD and OAuth2.
*.msftncsi.com	443	Used to test internet connectivity if the Power BI service can't reach the gateway.
*.microsoftonline-p.com	443	Used to authenticate the gateway app for Azure AD and OAuth2.
dc.services.visualstudio.com	443	Used by AppInsights to collect telemetry.

Tabla 4.1- Lista de FQDNs usados por la puerta de enlace.

Por defecto, para comunicarse con los servicios de Power Bi, la puerta de enlace utiliza el protocolo TLS (Transport Layer Security) 1.2.

TLS es un protocolo criptográfico que está basado en el SSL, creado en 1999 en un intento de agregar el protocolo HTTPS al navegador. La versión 1.2, que se sigue usando para la puerta de enlace a pesar de existir la versión 1.3, es el protocolo encargado de encriptar la mayoría de las comunicaciones que se producen en la red, como Gmail [10].

A esta puerta de enlace le puedes configurar el nombre de usuario con el que realizar las comunicaciones, para evitar así problemas de permisos. Además, una sola puerta de enlace se puede usar para todas las herramientas de Microsoft que la necesiten sin necesidad de configurar más.

Volviendo a el caso específico de Power Bi, una vez configurada la puerta de enlace, entra en efecto el tipo de conexión que se le haya configurado. Si se eligió importar, Power Bi te limita el número de actualizaciones diarias que se pueden realizar, mientras que en Direct Query los datos de actualizan con las peticiones que se realizan cada vez que se acceden a los informes que las contienen, ya que no están almacenados fuera de la base de datos.

4.2.4.- Exportación de datos

Power Bi es una herramienta de creación de informes en la web, por lo que, exceptuando el archivo .pbix que se genera con la aplicación de Power Bi desktop, no está diseñado para exportar los datos con consultas como una base de datos.

Sin embargo, recientemente se creó una extensión para Excel de Power Bi, como se ve en la figura 4.6, para exportar los informes creados en las hojas de Excel. Se creó para mostrar los informes diseñados en Excel, pero se puede usar también para extraer datos, como si se hiciese de una base de datos. Esta extensión es útil ya que es posible crear listas de datos nuevos en Power Bi desde otras herramientas de Microsoft y así podemos crear Excel con estos datos que no se podrían obtener de otra manera.

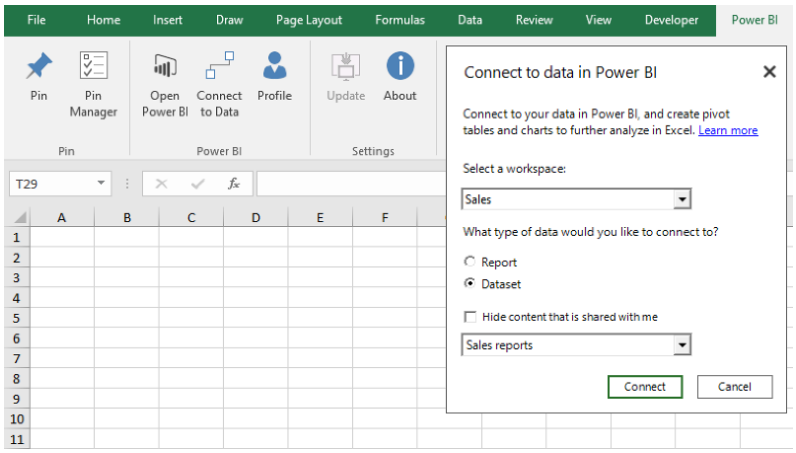

Figura 4.6- Extensión de Power Bi en Excel.

Pero, no está diseñada exactamente para extraer los datos en tablas normales, sino que las recoge en tablas dinámicas, como la de la figura 4.7, que tendremos que transformar en datos manejables.

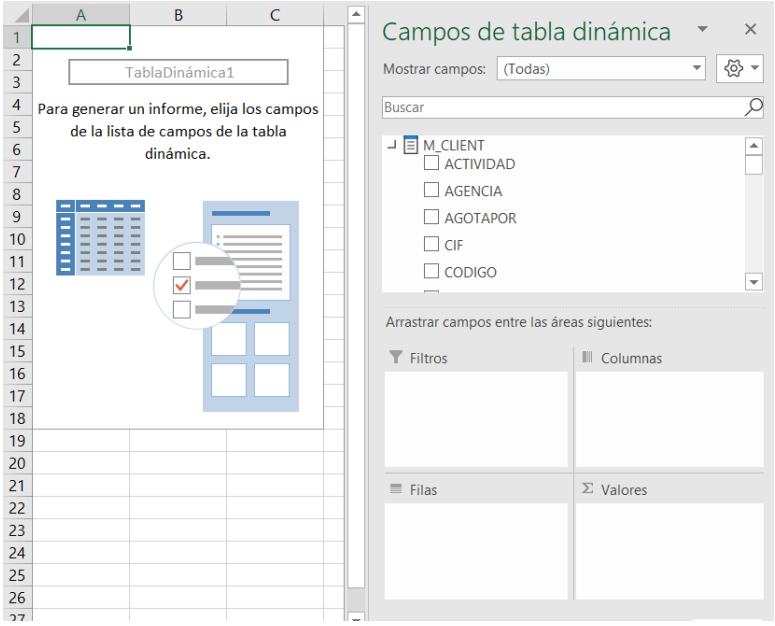

Figura 4.7- Tabla dinámica en Excel.

Esta conexión entre Excel y el informe, como con una base de datos, se produce mediante SSAS (Microsoft SQL Server Analysis Services), una vez conectada se crea un cubo OLAP, cuya representación teórica se puede ver en la figura 4.8, de datos que Excel puede moldear para representar los datos deseados mediante tablas dinámicas [11].

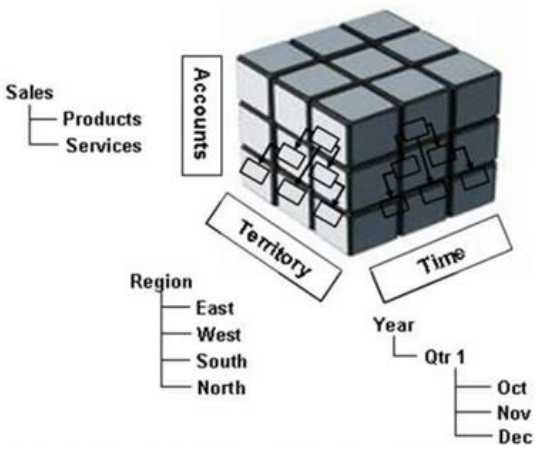

Figura 4.8- Cubo OLAP.

Un sistema OLAP permite mostrar de forma muy rápida una medida conjunta de datos, por ejemplo, el capital por año y provincia. Sin embargo, debido a esto, se ralentiza el análisis inicial cuanto más filtradas tengan que ser las salidas, por ejemplo, el capital por año en cada provincia y por cada código de vendedor.

4.3.- Power Automate

Power Automate, hasta hace poco llamado Microsoft Flow, es un motor de flujo de trabajo en la nube de Microsoft. Power Automate es una herramienta de programación diseñada para el uso general, integrando en la cuenta de Office 365, proporcionando una plataforma para fácilmente automatizar tareas sencillas y repetitivas debido a su bajo nivel de uso con un diseño basado en bloques. Además, proporcionar más ventajas cuanto mayor conocimiento de programación se tenga.

Power Automate funciona conectando bloques, como se puede ver en el ejemplo de la figura 4.9, llamados conectores. Existen conectores tanto de las demás herramientas de Microsoft, como Excel o Onedrive, como de otras empresas externas que añaden sus propios conectores. Esto ayuda en casos donde la base de datos no es un SQL de Microsoft, sino que es otro tipo distinto de base de datos, como PostgreSQL.

Estos conectores son herramientas que permiten unir dos o más aplicaciones, que pueden ser independientes, y automatizar fácilmente procesos entre ellos. Como coger datos de una base de datos y mandarlos por correo con Outlook.

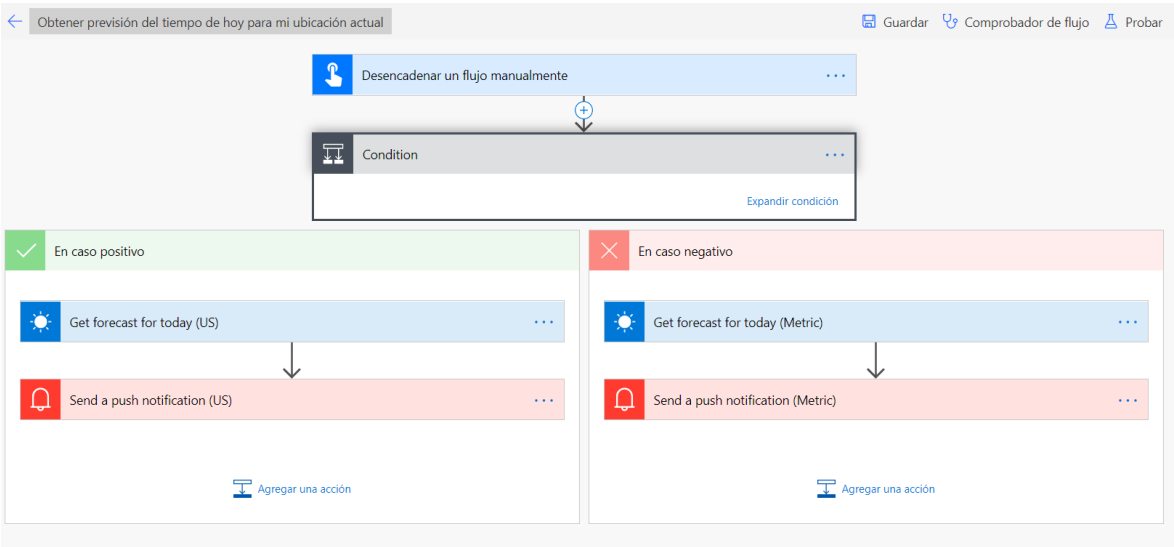

Figura 4.9- Ejemplo de flujo.

Como se puede imaginar por el estilo visual de la figura anterior, y como se comentó anteriormente, Power Automate en una herramienta que te permite realizar con facilidad procesos sencillo, lo que permitiría a personas no expertas aliviar trabajo con facilidad. Pero está muy limitado a las funciones de los conectores, ya que si no existe un conector para la función que necesitas hay que programarlo o buscar una alternativa.

En cuanto a las licencias, la licencia de Office 365 te permite usar los conectores 'premium' de la herramienta. Sin embargo, para poder usar la puerta de enlace, la misma que se instala para Power Bi, para los conectores que la requieren se necesita una licencia de Power Automate Premium, que son 15€ por usuario. Pero es posible realizar, con la cuenta

de solo un usuario, automatizaciones que hagan referencias a otras cuentas de Microsoft, quitando la necesidad de contratar más licencias mientras se trabaje dentro de los límites de esta.

4.4.- Powershell

Y al ultima herramienta de Microsoft que se usó para el proyecto es Powershell, que, aunque viene incluido con Windows y no con el paquete de Office 365, es de gran utilidad para realizar montajes automáticos en el servidor.

Powershell es un shell de línea de comandos y un entorno para la automatización de tareas basado en el Framework .NET, perfecto para la automatización de procesos. Este, como ya se comentó, viene integrado con Windows 10 y Windows Server, pero se puede obtener para versiones anteriores y Linux.

Powershell no es un lenguaje de programación compilado como C, si no que funciona por scripts, contando con un conjunto de comandos preparados para ser usados por el usuario. Entre la lista de comandos existentes hay para SQL servers y Excel, lo que permite manejar los datos con los que queremos trabajar eliminando la necesidad del usuario para, por ejemplo, ejecutar las macros de los Excel que se habían necesitado con VBA.

5.- ARQUITECTURA INICIAL

En este proyecto se trabajó con clientes que ya tenían funcionando, de una forma u otra, una base de datos donde han ido acumulando la información de su negocio. Pero que no la han estado explotando para obtener informes en vivo del estado de su empresa, o que el método que han estado usando no es lo suficientemente automático como para resultar eficiente, requiriendo grandes cantidades de horas 'perdidas' por trabajadores de la empresa de forma periódica.

Todas estas empresas presentan informaciones muy diversas entre ellos, una empresa puede tener una estructura interna completamente diferente a de la empresa anterior, lo que complica la generalización del proceso. Además, la diferencia de los resultados que se esperan obtener con las automatizaciones acaba generando proyectos enormemente personalizados donde lo único similar es la forma de obtención de los datos a analizar.

Por eso, lo primero que se debe analizar es la jerarquía con al que se va a trabajar: los servidores existentes, el tipo de base de datos, su localización, el resultado que se quiere tener.

Ya que todas estas variables pueden afectar enormemente al montaje final del proyecto, desde el tipo de base de datos hasta donde se encuentra físicamente:

5.1.- Bases de datos.

Una base de datos es, principalmente, un conjunto de datos que pertenecen a un mismo contexto y que están almacenados sistemáticamente para su posterior uso.

En cuanto a decidir qué tipo de base de datos, la decisión es generalmente entre las racionales y las racionales, llamadas respectivamente SQL (Server Conection String) y NoSQL, representadas en su composición más básica en la figura 5.1.

Figura 5.1- . SQL vs NoSQL.

Aunque ambas son opciones viables, existen diferencias claves entre ambas que hay que tener en cuenta a la hora de elegir.

5.1.1.- SQL database

Un servidor SQL es un sistema de gestión de bases de datos racionales (RDBMS), que, en su nivel más fundamental, almacena información en tablas. Estas tablas son los objetos de las bases de datos que se comportan como contenedores para los datos, organizándolos lógicamente en un formato de filas y columnas.

Es decir, cada fila es una entidad que se describe con las columnas, las cueles contienen sus atributos. Por ejemplo, si se crea una tabla de clientes, cada fila de ella sería un cliente y su información se describiría en las columnas que tenga esa tabla: como el nombre en la columna CustomerName o la dirección del cliente en la columna CustomerAddress.

Las filas de las tablas no tienen un orden específico, por lo que, si se quiere especificar uno, se deberá hacerlo en la consulta que se realice de ella.

Las tablas de la base de datos se pueden usar también como mecanismo de seguridad, proporcionando permisos de acceso a los usuarios del servicio a nivel de tabla.

En cuanto a la base de datos que elegir para un montaje real, las racionales son las más comúnmente usadas, como PostgreSQL o Microsoft SQL Server.

Las tablas de Microsoft SQL están, a su vez, contenidas en objetos denominados esquemas, que también tienen la función de trabajar como límites de seguridad, dividiendo la base de datos en niveles de esquemas y así separar a los usuarios en niveles de acceso. Los esquemas pueden contener hasta 2.147.483.647 tablas con hasta 1024 columnas cada una.

Cuando se diseña una tabla de una base de datos, las propiedades que se les asigne y las columnas creadas dentro de las tablas controlaran los tipos de datos permitidos y el rango de datos que se aceptaran. Un diseño adecuado de una tabla hará que sea más fácil y rápido almacenar datos y recuperar los datos de ella.

Físicamente, las tablas de un servidor SQL se almacenan en una base de datos como páginas de 8 KB de tamaño. Estas páginas se almacenan de forma predeterminada en una sola partición junto con el grupo de archivos predeterminados primarios. Además, una tabla se puede almacenar en múltiples particiones, en las que cada grupo de filas se almacenan en varias particiones en función del contenido de una columna específica, como se representa en la figura 5.2.

En cuanto a la escalabilidad, en casi todas las situaciones una base de datos SQL es verticalmente escalable, es decir, que se puede aumentar la carga en un solo servidor aumentando sus características, como RAM, CPU o añadiendo más memoria SSD.

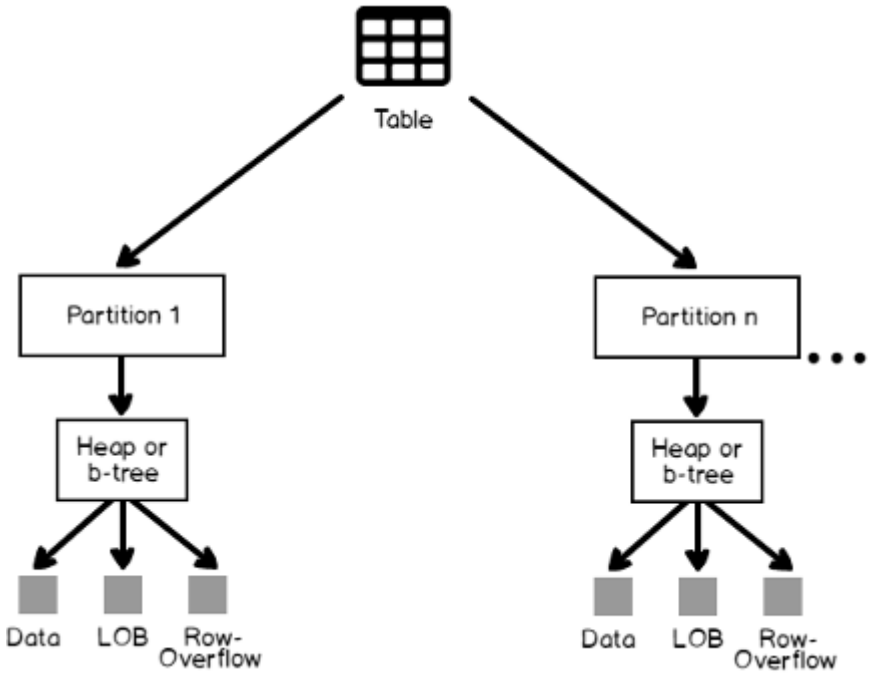

Figura 5.2- Estructura de una base de datos SQL.

Las bases de datos SQL cuentan con una gran cantidad de soporte, tanto por los proveedores como por la comunidad, para la mayoría de las herramientas que trabajan con análisis de datos.

Dentro del .NET Framework, un software framework desarrollado por Microsoft, existen objetos llamados 'Connections'. Estos se heredan de la clase 'DbConnection' con una propiedad 'ConnectionString' para cada proveedor de bases de datos, para poder usarla desde cualquier computadora donde este instalado. La sintaxis de la cadena de conexión especifica de cada proveedor está incluido en la propiedad 'ConnectionString'.

En el caso de Microsoft Server SQL es llamada 'System.Data.SqlClient', que contiene las clases para interactuar con el origen de datos.

5.1.2.- NoSQL database

Una base de datos no relacional, también llamada NoSQL, es aquella que utiliza un esquema distinto al de filas y columna que se vio en las bases de datos SQL, considerado el esquema de bases de datos tradicional. En su lugar, las bases de datos no relacionales usan un esquema personalizado que esta optimizado para el tipo de datos que se van a almacenar, y sus requisitos específicos. Por ejemplo, los datos se pueden almacenar como documentos JSON, como pares claves o como un grafo que se compone de bordes y vértices.

Como indica su nombre, el termino NoSQL hace referencia a que el almacén de datos no utiliza SQL, sus consultas, en su lugar, usan un lenguaje de programación y métodos distintos para consultar los datos. Pero, a pesar de ser una base de datos no relacional, admite consultas compatibles con SQL y la estrategia de ejecución de consultas es, normalmente,

similar a la que una RDBMS tradicional ejecutaría esa misma consulta en una base de datos SQL.

Una base de datos NoSQL utiliza un esquema dinámico para manejar datos no estructurados. Estos datos se almacenan de varias maneras, bien sea orientados a documentos, columnas, gráficos u organizados, como un almacén KeyValue, que consiste en un conjunto de registros que, a su vez, contienen diversos campos con datos. Esta flexibilidad permite crear documentos sin tener que crear estructuras definidas primero. Además, cada uno de los documentos podrá tener una estructura propia y única. En cuanto a la sintaxis, varia de una base de datos a otra, y permite agregar campos a medida que se avanza con más facilidad que con una base de datos SQL, que requeriría programar código para hacerlo.

Las bases de datos NoSQL son escalables horizontalmente, viéndose la diferencia con las bases de datos SQL en la figura 5.3. Esto significa que se maneja más tráfico al fragmentar o agregar más servidores a la base de datos NoSQL. En consecuencia, una base de datos NoSQL puede escalarse enormemente, siendo la mejor opción para grandes conjuntos de datos y en constante cambio.

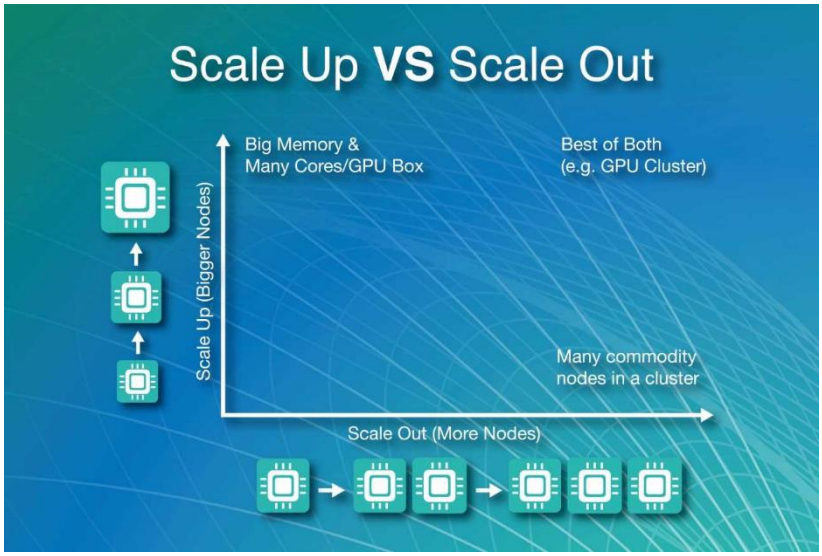

Figura 5.3- Crecimiento horizontal vs vertical.

Sin embargo, para algunas bases de datos NoSQL se debe confiar en el soporte de la comunidad debido al limitado soporte de expertos externos disponibles para configurar e implementar una base de datos a gran escala.

En resumen, las razones por las que se eligió una base de datos SQL sobre una NoSQL son:

- Como el objetivo del proyecto es crear consultas e informes complejos, se requiere crear un script que recupere y presente los datos deseados. NoSQL no admite relaciones entre tipos de datos, limitando las posibilidades ya que en las

bases de datos se suele hacer referencia a otras tablas codificando el dato al que hace referencia. Por ejemplo, si existe un cliente llamado Cliente01 codificado con el número 001 en la tabla de clientes, donde se incluye el resto de información de este, en la tabla de facturas, cuando hay una factura de este cliente en vez de poner toda la información del cliente se pone su código 001, haciendo muy útil las relaciones entre tablas.

- SQL es más adecuado para transacciones pesadas o complejas debido a que es más estable y garantiza la integridad de los datos.
- Si no se esperan muchos cambios en los datos, ni se están trabajando con un gran volumen de datos de distintos tipos no se estarían aprovechando las ventajas de NoSQL.
- Las limitaciones con los Querys de datos y las diferencias de soporte proporcionado por la comunidad de las bases de datos NoSQL generarían muchos más problemas para montajes muy personalizados.

Sin embargo, eso no quiere decir que SQL sea la mejor solución en todas las situaciones, depende de la información con la que se trabaje. NoSQL es muy flexible, lo que permitiría añadir nuevas funciones y tipos de datos con facilidad.

5.2.- Localización de la base de datos

La localización física de la base de datos es importante ya que el nivel de control que se tiene sobre la información de la base de datos podría limitar las opciones de analizado, por lo que se pueden encontrar 3 situaciones representadas en la figura 5.4.

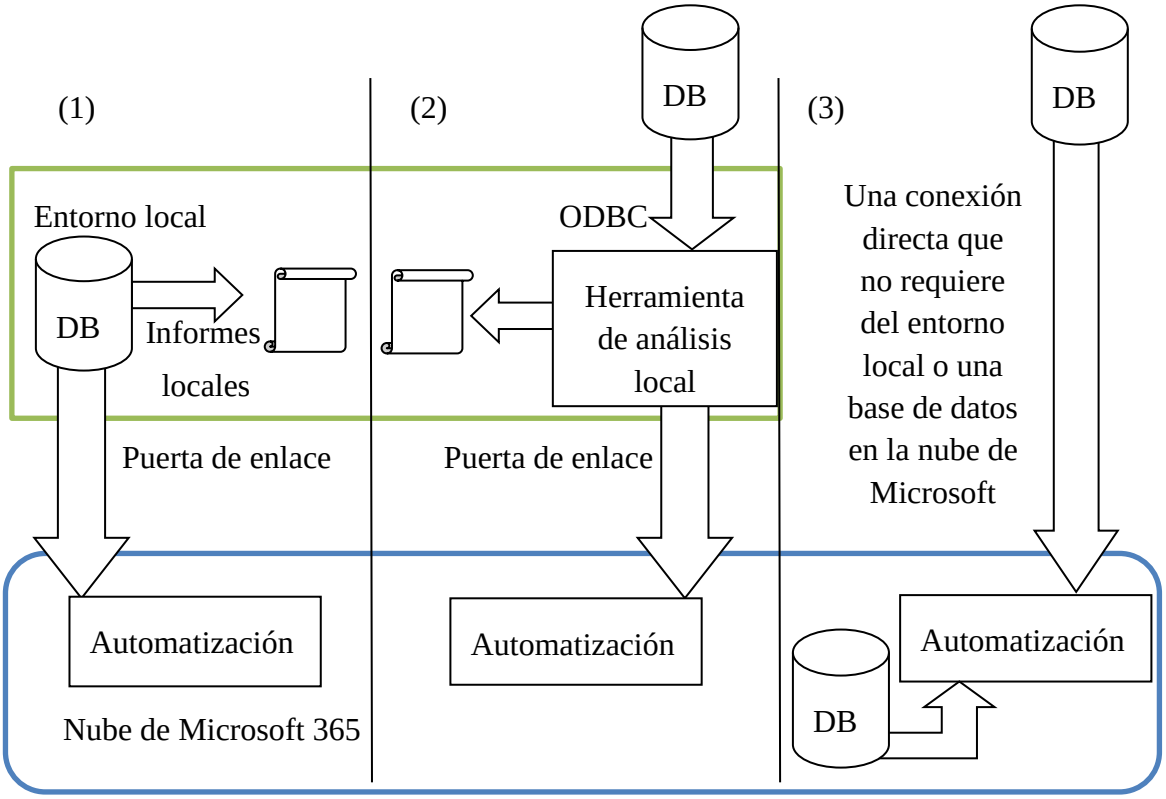

Figura 5.4- Comparación superficial de diversos montajes de automatización.

En resumen, la base de datos se puede encontrar en un entorno local, donde se tiene un mayor control sobre ella, permitiendo instalar una puerta de enlace, conectando directamente las herramientas de Microsoft con la base de datos, la situación 1. Y donde la base de datos se contrató a una empresa externa que proporciona un espacio dentro de sus servidores para dar un servicio 'en la nube', situación 2 y 3. Dentro de la segunda posibilidad existen otras dos opciones: que el servicio de nube sea proporcionado por Microsoft, con Microsoft Azure, u otra empresa con buen soporte con Microsoft, situación 3, lo que permitiría conectar las herramientas de Microsoft con la base de datos sin requerir ningún tipo de montaje en el servidor local y que la empresa de terceros no cuente con un buen soporte, ni de expertos ni de la comunidad, lo que obligará a realizar un montaje más complejo para poder acceder a la información deseada, situación 2.

De estas situaciones la 2 es la menos deseada, variando el grado de complejidad dependiendo de la empresa a la que se le haya contratado el servicio. Ya que depende de que se hayan desarrollado controladores, o drivers, para poder conectarte a los datos, bien sea por la comunidad o por el propio proveedor.

Por ejemplo, si se tiene contratado un servidor de Microsoft Azure las herramientas de Microsoft tienen conectores propios construidos que facilitan el acceso y el manejo de datos.

En el caso de no tener ninguna de conectarse a la base de datos de forma normal se usaría el programa ODBC (Open Database Connetivity), un estándar de acceso a las bases de datos desarrollado por SQL Access Group (SAG) en 1992. El objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, independientemente del sistema de gestión de bases de datos (DBSM) en el que se almacenen dichos datos. ODBC logra esto al insertarse en la capa intermedia denominada nivel de Interfaz de Cliente SQL (CLI), entre la aplicación y el DBMS [12], como se ve en la figura 5.5. El propósito de esta capa es traducir las consultas que realizan las aplicaciones en comandos que la base de datos entienda.

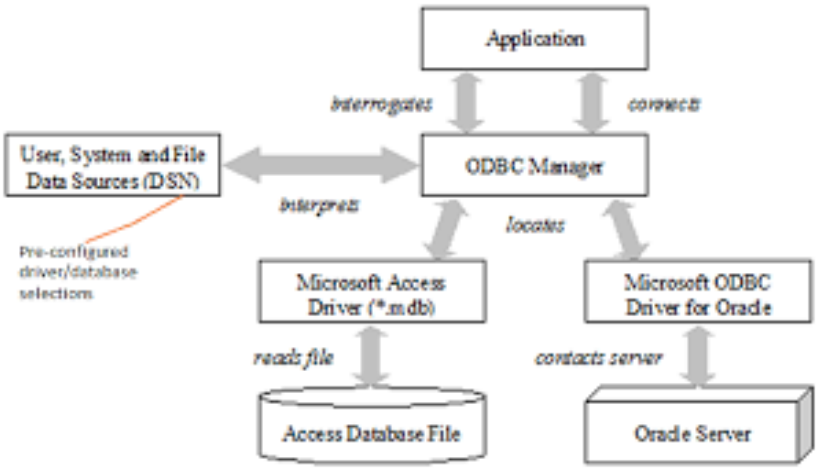

Figura 5.5- Situación del ODBC.

Sin embargo, si se requiere el uso del ODBC para obtener los datos de la base de datos, porque no existe soporte para Microsoft del proveedor, surge el problema de que, no existir actualmente un conector de la nube de las aplicaciones de 365 para ODBC. Entonces, se requiere de otra aplicación distinta para obtener los datos del ODBC, para luego llevarlo a la nube de Microsoft, alargando la cadena de conexiones para realizar una consulta a los datos para la automatización.

En este caso, se puede usar el Query de Excel para obtener los datos, ya que cuenta con un driver de ODBC. Y así, además de poder conectar los datos con la puerta de enlace en el servidor, se puede aprovechar que los datos se encuentran en el Query de Excel para realizar un montaje de informes locales que se pueda distribuir entre los usuarios del servidor o los miembros de la empresa que tenga acceso a carpetas compartidas en el servidor.

6.- AUTOMATIZACIÓN

Debido a la enorme variedad entre los datos de cada empresa, como ya se comentó antes, el análisis de dichos datos produce informes muy distintos de unos a otros. Sin embargo, los procedimientos a realizar para llegar a poder analizar los datos pueden ser muy similar entre ellos, por lo que, en este proyecto, se va a centrar en preparar el terreno para realizar la presentación de los informes con los datos analizados

Para el proceso de automatización se preparó un escenario que incluye la arquitectura más común encontrada a lo largo de los proyectos de automatización en empresas realizados: un servidor SQL dentro de uno de los servidores físicos con Windows Server de la empresa. Al que se tiene acceso constante gracias a una VPN (Virtual Private Network) [13] y con un usuario administrador propio que ningún otro usuario va a usar.

Para aprovechar al máximo todas las aplicaciones proporcionadas por Microsoft se van a realizar una variedad de procesos:

- Se va a utilizar Excel en el servidor para generar documentación local para los trabajadores de la empresa.
- Se va a aprovechar Outlook para enviar correos diarios con algunos de los informes generados en el punto anterior.
- Se va a aprovechar Power Automate para automatizar la gestión de correos y reenviarlos a los trabajadores correctos.
- Se va a utilizar Power Bi para crear un análisis masivo de datos, explicado en el apartado de Big Data Análisis, para que los trabajadores puedan consultar el estado de los trabajos desde la web.

En cuanto a la licencia de Microsoft 365: se usó la licencia de Microsoft 365 E3, de la tabla 2.1, ya que es la mínima para contar con Office y un servicio Exchange para cada usuario. Se necesitan otras licencias adicionales para realizar algunos montajes, pero se explicará el porqué de su necesidad cuando se tenga el contexto de su uso.

Definidas las bases, el montaje final del proceso será el representado en la figura 6.1:

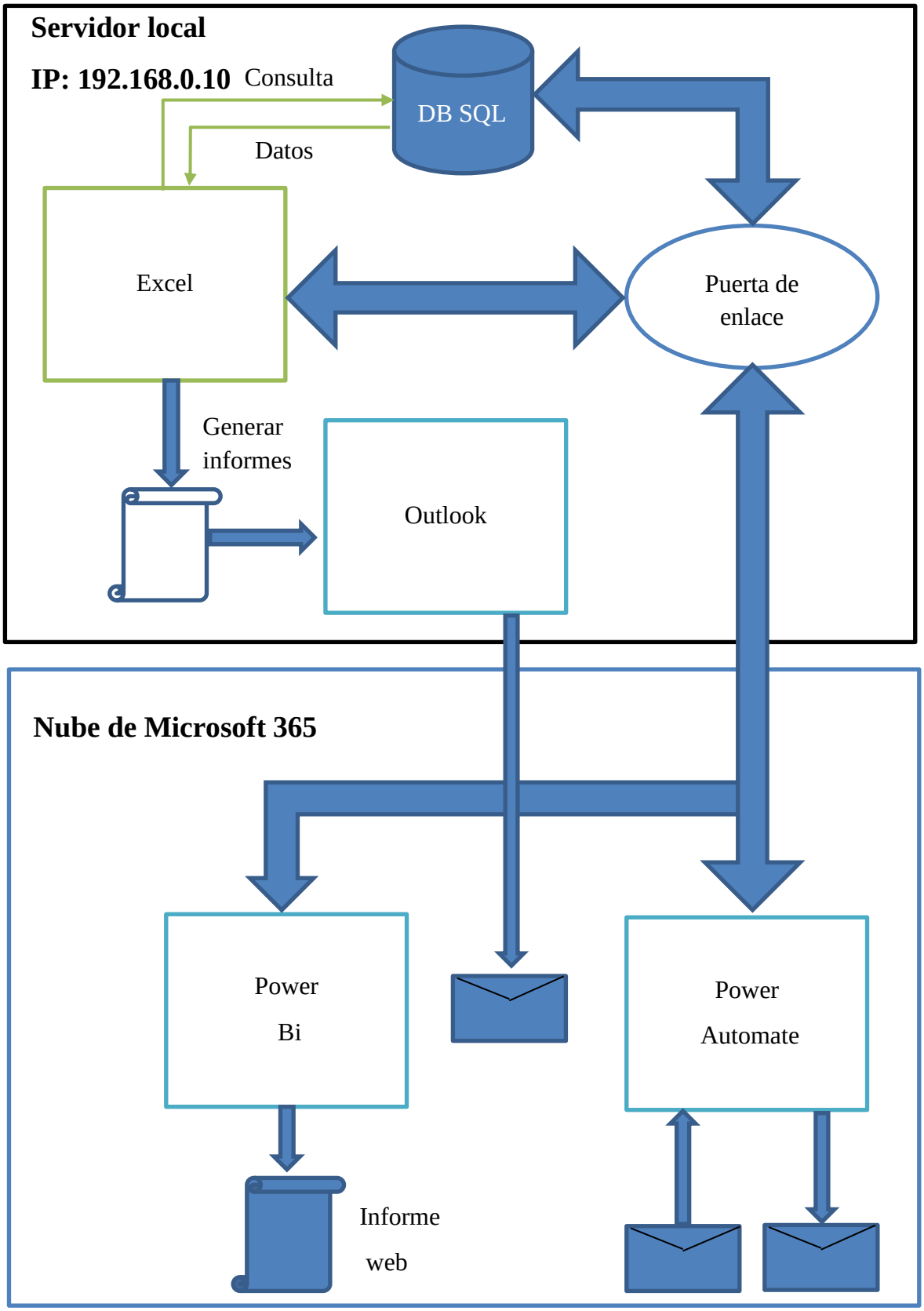

Figura 6.1- Montaje final del proyecto.

6.1.- Análisis Bi de datos

Para conectarse a la base de datos con Power Bi, como ya se explicó, es recomendable descargar Power Bi Desktop. El cual te permite más libertad a la hora de conectarse a las bases de datos y te da acceso al Query de Power Bi, con el que se podrá tratar mejor los datos obtenidos.

El objetivo de este apartado es generar un informe web que permita a los trabajadores comprobar el listado de trabajos abiertos e interactuar con ellos para obtener información detallada de su estado.

Para la conexión con la base de datos, debido a que está situada en el mismo servidor desde el que se está trabajando, no requiere ninguna cadena de conexión, el conector la detectara automáticamente al introducir la IP del servidor. Si se hubiese necesitado levantar una conexión externa, como ODBC, para poder acceder a los datos, se habría necesitado una cadena de conexión.

```
Driver={PostgreSQL};Server=IP address;Port=5432;Database=myDataBase;Uid=myUsername;Pwd=myPassword;
```

Este ejemplo es para conectarse a una base de datos PostgreSQL externa a la que se ha levantado un ODBC para conectarse. Además, habría que tener en cuenta el estándar con el que se está enviando, como por ejemplo ANSI, ya que habría que especificarlo dentro de la parte de Driver del código anterior. Y, si se intenta convertir a un estándar manejable una columna de datos que no sea una cadena de datos, como un archivo json, la cadena de conexión produciría error intentando acceder a ella. Para evitar errores de este tipo es posible añadir solo las columnas deseadas:

```
SELECT column1, column2, ...
FROM table_name;
```

De esta forma reducimos la carga de datos de la consulta y evitamos los problemas de las columnas incompatibles con el análisis de datos.

Una vez conectado a la base de datos, debemos elegir las tablas que necesitamos para llevar a cabo el informe. Para este caso se usaron las tablas que muestran las ordenes, los presupuestos de los que nacen dichas ordenes, la información de los clientes, las tareas introducidas por los trabajadores para saber que se ha hecho con el trabajo y la lista de albaranes para saber si el trabajo ya se ha enviado al cliente.

Si se ha realizado la conexión la conexión por Direct Query en vez de Importarlo, dependiendo de las necesidades explicadas anteriormente, no sería posible modificar la conexión con el Query, por ejemplo, filtrar los datos recibidos. Pero será posible trabajar con ellos en los siguientes pasos del proyecto.

A continuación, es necesario crear las relaciones entre las tablas, como en la figura 6.2, para poder mostrar la información filtrada correctamente. Gracias al sistema con el que montan los servidores SQL, las relaciones ya están bien definidas con los datos de las columnas, por ejemplo, la tabla de ordenes tiene una columna que hace referencia al código del presupuesto y del cliente al que pertenece.

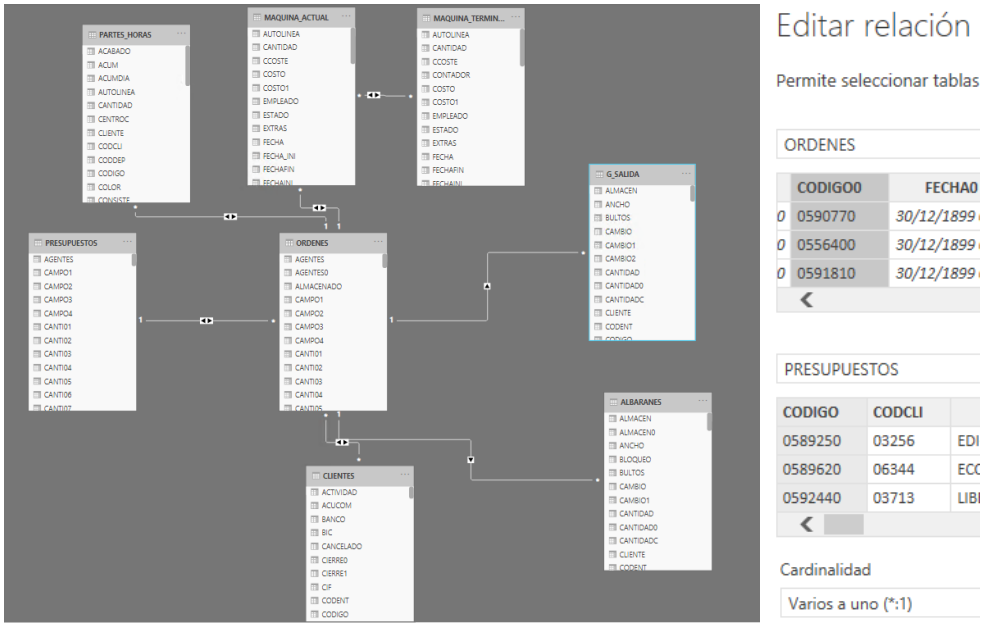

Figura 6.2- Relaciones entre tablas en Power Bi.

Una vez creadas las relaciones correctamente, Power Bi relacionará de forma rápida los datos que introduzcas en las tablas, como en figura 6.3 que es una captura de un informe simple, o las gráficas del informe. Pudiendo combinar rápidamente toda la información de todas las tablas, aunque haya múltiples coincidencias, como varios presupuestos con el mismo cliente.

COD. PRESUPUESTO	COD. ORDEN	CANTIDAD	DESCRIPCIÓN	IMPORTE	REP.	FECHA ENTREGA ORDEN	FECHA EMISIÓN PRES.	FECHA APER. ORDEN
0503670	0023266		[redacted]			30/12/1899 0:00:00		09/06/2017 10:08:00
0503670	0023283					30/12/1899 0:00:00		13/06/2017 19:08:00
0503670	0023445					30/12/1899 0:00:00		01/08/2017 8:33:00
0503670	0023524					30/12/1899 0:00:00		28/08/2017 17:09:00
0503670	0023602					30/12/1899 0:00:00		19/09/2017 18:41:00
0503670	0023694					30/12/1899 0:00:00		10/10/2017 15:05:00
0503670	0023757					30/12/1899 0:00:00		30/10/2017 12:42:00
0503670	0023911					30/12/1899 0:00:00		29/11/2017 18:40:00
0503670	0023912					30/12/1899 0:00:00		29/11/2017 18:42:00
0503670	0024041					30/12/1899 0:00:00		02/01/2018 13:24:00
0503670	0024121					30/12/1899 0:00:00		22/01/2018 17:20:00
0503670	0024284					30/12/1899 0:00:00		05/03/2018 19:28:00
0503670	0024380					30/12/1899 0:00:00		26/03/2018 18:46:00

Figura 6.3- Tabla usando las relaciones.

Como se quiere que el informe de mantenga actualizado constantemente, ya que se necesita saber el estado en vivo del trabajo, se usó Direct Query, lo que impidió utilizar las

funciones del Query de Power Bi. Sin embargo, existen otras funciones de Power Bi que te permiten manejar los datos después de obtenerlos de la consulta, no reduciendo la carga de datos de ella, pero permitiéndote manejar los datos igualmente. Es posible crear nuevas columnas, que, aunque no se aplican a la consulta realizada a la base de datos, se aplican constantemente en el informe de Power Bi. También se pueden crear medidas, que no es lo mismo que una columna nueva, estas calculan un resultado a partir de unas fórmulas de expresiones de análisis de datos (DAX), en resumen, son fórmulas de cálculos parecidas a las de Excel, como SUM.

Una vez añadida la tabla con los datos de las ordenes activas se pueden añadir filtros con los que el usuario que vea el informe puede interactuar, como los de figura 6.4, para poder elegir la orden de la que ver la información detallada.

Figura 6.4- Objeto filtros de Power Bi.

Con este objeto visual podemos permitir a los trabajadores interactuar con el informe. Cada usuario que acceda al informe podrá modificar solo los filtros de su visualización, por lo que dos trabajadores pueden acceder a la vez sin modificarse la visualización entre ellos.

En este informe se incluyeron también otras páginas en las que se incluyen gráficos de ventas y otras informaciones privadas de la empresa. Es por esta razón que el proyecto tiene un enfoque tan general, solo se enseña el proceso para llegar al análisis y se están explicando solo los detalles más importantes del proceso.

Power Bi está diseñado para crear páginas de informes que actúan únicamente en ellas mismas y de forma muy limitada con las demás páginas del informe. Es por lo que actualmente no hay forma de, por ejemplo, seleccionar con el ratón un cliente de la lista y que te lleve a otra página donde te muestre la información detallada del mismo. Sin embargo, existen métodos que simulan esta situación, como Drillthrough y una forma de revelar datos una vez filtres en la página.

DrillThrough te permite ir a otra página del informe llevándote como filtro el dato sobre el que estés. De esta forma, con el botón derecho del ratón, como se ve en la figura 6.5, podemos simular el mostrar la información del cliente llevándonos como filtro el código del cliente a otra página donde se mostrará la información deseada.

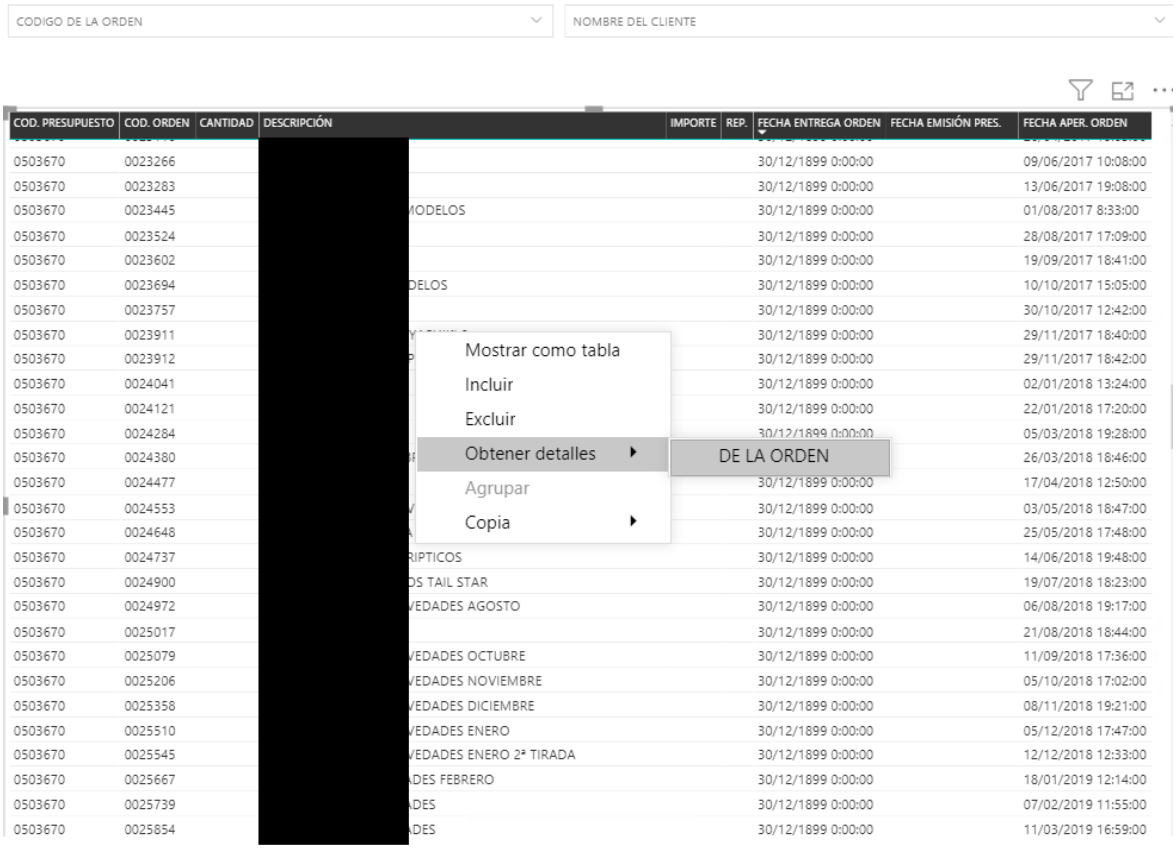

Figura 6.5- DrillThough de Power Bi.

Por ejemplo, en este caso, llamado a una página oculta 'DE LA ORDEN' se está dando a entender que pulsando ese botón podrás obtener información de la orden sobre la que pinchasteis. Pero en realidad significa que estas yendo a la página "DE LA ORDEN" con el código de orden como filtro, llegando así con el filtro COD_ORDEN = 0023694, que, si está bien diseñadas las relaciones entre tablas, te mostrara únicamente la información relacionada con ese código de orden, como se ve en la figura 6.6.

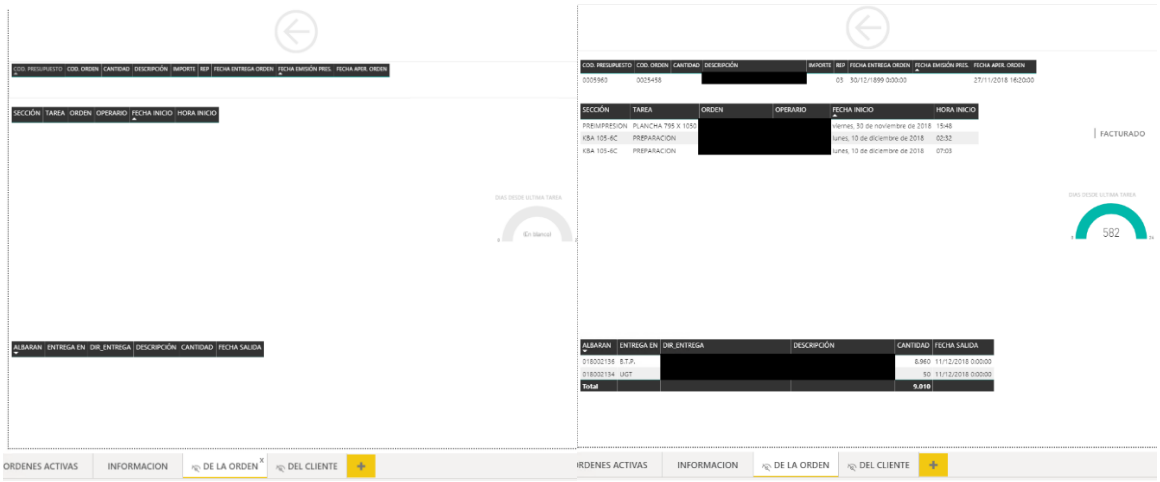

Figura 6.6- Informe con (derecha) y sin (izquierda) aplicarse el drillthough.

La segunda opción es crear un sistema que, cuando se elija una orden en un filtro, como el mostrado en la figura 6.4, se revele alguna parte del informe donde se representen gráficos o se muestren datos que serían innecesarios si se estuviesen aplicando a más de un código de orden. Para ello utilizaremos la función de añadir nuevas medidas a las tablas que tenemos que nos proporciona Power Bi.

Las medidas, como ya se explicaron, actúan aplicando una función a todas las filas de la tabla que se están mostrando en el informe en ese momento. Por eso, si creamos una función que reaccione al filtrado de la tabla podremos crear el desencadenante para mostrar el resto de los datos.

```
SlicerCheck = if(ISFILTERED(ORDENES[CODIGO]),1,2)
```

Con esta sencilla formula se creó un valor 'SlicerCheck', que será 1 cuando se esté filtrando el código de las ordenes de cualquier forma, consiguiendo así el desencadenante para mostrar el resto de la información. Si añadimos esta función a la página mostrada en la figura 6.5 podemos escondes detrás de la tabla el resto de información de la orden para que cuando se introduzca un filtro se muestre. Para ello hay que crear una nueva medida para hacer transparente el fondo de la tabla para así ocultar el resto cuando no haya filtro y mostrarlo cuando lo haya.

```
Make Transparent =
    IF(
      ISFILTERED(ORDENES[CODIGO]),
      "#FFFFFF00",
      "White"
    )
```

Con esta modificación de la medida anterior tenemos el código que devuelve los colores que aplicar al fondo de la tabla, que cambiará de valor automáticamente cuando se produzca un filtrado. El código hexadecimal #FFFFFF00 representa al color transparente, lo que nos permite hacer que se vea lo que hay detrás de la tabla de órdenes, como se representa en la figura 6.7.

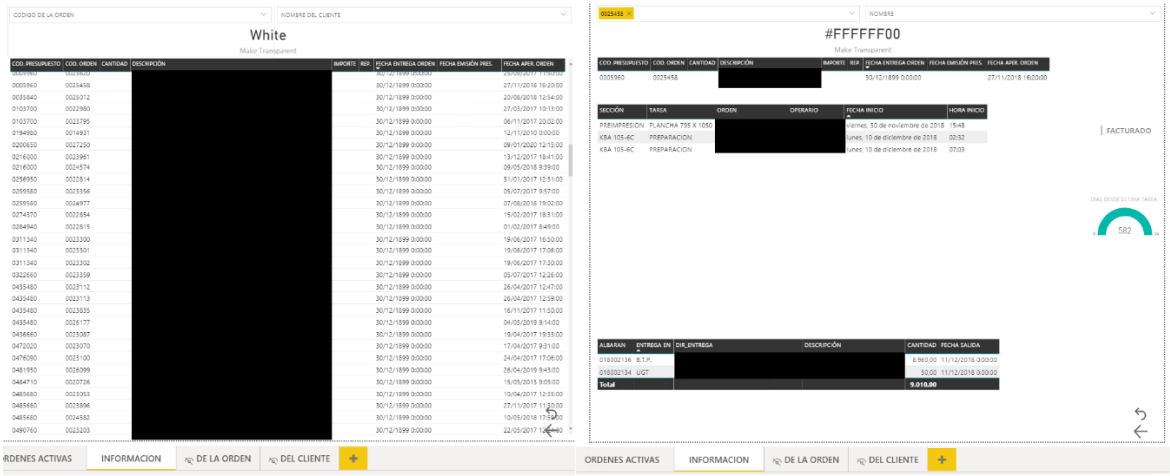

Figura 6.7- Informe sin (izquierda) y con (derecha) filtro.

Una vez realizado el informe con la información deseada hay que publicarlo en tu grupo de Power Bi en la web, desde donde puedes decidir donde publicarlo: en una página web, en Sharepoint, etc. El problema viene cuando quieres decidir quién es capaz de verlo. Si es un informe público para que vea cualquier cliente, como las valoraciones de la empresa o datos que se quieren compartir, no hay problema, Power Bi te proporciona un iframe para poner en cualquier HTML que te permitirá acceder al informe con facilidad.

```
<iframe width="800" height="600"
src="https://myserver/reports/powerbi/Sales?rs:embed=true" frameborder="0"
allowFullScreen="true"></iframe>
```

Si se quiere restringir el acceso a los miembros de la empresa se puede obligar a iniciar sesión con una cuenta de Microsoft que tenga permiso de Miembro en el informe. Sin embargo, esto requiere una licencia de Power Bi Pro para cada usuario que requiera acceso al informe, aumentando el precio del montaje en 8€ al mes por usuario que acceda al informe.

6.2.- Generación de informes periódicos

Aunque Power Bi permite manejar de forma fácil y rápida grandes cantidades de datos, aún no tiene ningún tipo de forma solida de generar archivos físicos, como un PDF. Para estas situaciones es mejor utilizar otras herramientas de Microsoft que te proporcionan una mayor flexibilidad.

En este caso se usó Excel para recoger los datos y, moldeándolos, crear varios informes personalizados. Estos informes, aunque muy distintos en su aspecto final y el tipo de contenido, requieren un montaje inicial idéntico entre ellos solo cambiando los datos que se recogen para mostrar en el informe. Es por esta razón que solo se mostrará una de ellas, pero explicare todas las diferencias importantes que hayan surgido en los demás montajes. Además, como ya comenté antes, contienen información privada de las empresas por lo que hay algunas características que se pudo haber omitido.

Como con Power Bi, Excel cuenta con la opción de realizar consultas directas con la base de datos para obtener los datos deseados. Sin embargo, Excel es mucho más sensible al formato de las columnas y, además, como traduce los datos de la consulta al formato de la hoja de cálculos tras cada actualización. Excel tiene un tiempo de carga mayor que Power Bi, por lo que es recomendable cargar únicamente las columnas con las que se van a trabajar.

```
SELECT [TABLA].[COLUMNA1], [TABLA].[COLUMNA2], …
 FROM [DATABASE].[TABLA] WHERE [‘CONDICIÓN DE FILTRADO’]
```

Este proceso añade una nueva página a la hoja de cálculo con una tabla conteniendo los datos de la consulta, hay que tener en cuenta que Excel tiene un número máximo de filas de 1.048.576 filas, por lo que más datos que el límite de filas no se cargarán. Tras eso se puede utilizar como una tabla de Excel normal para hacer referencia a su contenido.

Si se quieren representar algún tipo de suma de valores, como por ejemplo la suma de las facturas mensuales de un representante, es preferible utilizar tablas dinámicas. Para ello, a pesar de añadir un paso más de conexión en la consulta, se utilizó la extensión de Power Bi para Excel, ya que la posibilidad de crear medidas con facilidad y el poder comprobar la veracidad de las modificaciones antes de publicarlas en Power Bi desktop ayudan en el montaje a futuro.

Además, trabajando con tablas dinámicas, Excel cuenta con comandos para acceder a los datos de las medidas aplicando filtros en el propio código, permitiendo variar el filtrado de forma activa, como por ejemplo la fecha actual, para representar los datos adecuados en cada momento.

```
=SI.ERROR(
        IMPORTARDATOSDINAMICOS(
                "[Measures].[TIPO_C]";
                Visitas_V!$A$3;
                "[VISITA].[Mes]";
                UNIRCADENAS(;;"[VISITA].[Mes].&[";MES("1"&"/"&B$3&"/"&"
                1");"]");
                "[VISITA].[REPRE]";
                UNIRCADENAS(;;"[VISITA].[REPRE].&[";$O$4;"]");
        0
)
```

Por ejemplo, este es un código usado en uno de los Excels cuyo objetivo es representar el número de visitas que un representante ha realizado a clientes cada mes del año, sacando

la información de una tabla dinámica. Excel cuenta con una fórmula, IMPORTARDATOSDINAMICOS('Tabla','Filtro1','Filtro2'), para obtener la información directamente de la tabla dinámica que te devuelve el valor de la medida seleccionada que cumpla con los filtros deseados.

Si tienes que hacer referencias a una variable, como por ejemplo el código de representante o la fecha a filtrar, como el código te obliga a poner [TABLA].[COLUMNA].[VALOR A FILTRAR] la mejor manera de hacerlo automático es usando el comando UNIRCADENAS(), como se ve en código mostrado.

Además, si intentas mostrar una medida con filtros que no existen, por ejemplo, el número de visitas en junio estando en mayo, Excel devolverá un error #Value. Para solucionarlo es recomendable utilizar el comando SI.ERROR() para ponerlo a 0 si la salida fuese un error, poniéndolo a 0 siempre que la medida no existiese, quedando la representación de la figura 6.8 con datos de una tabla de ejemplo.

ACTIVIDAD COMERCIAL - REPRESENTANTE1 - AÑO 2020												
	ENERO	FEBRERO	MARZO	ABRIL	MAYO	JUNIO	JULIO	AGOSTO	SEPTIEMBRE	OCTUBRE	NOVIEMBRE	DICIEMBRE
Nº CONTACTOS CARTERA	10	5	12	0	0	0	0	0	0	0	0	0
Nº CONTACTOS POTENCIALES	50	20	30	0	0	0	0	0	0	0	0	0
Nº TELECONTACTOS	60	130	110	10	50	70	26	0	0	0	0	0

Figura 6.8- Ejemplo aplicación del código de tablas dinámicas

Con esto ya se tiene acceso desde Excel a todos los datos de la base de datos y cualquier tipo de información que solo, por ejemplo, los datos de las respuestas de encuestas de satisfacción que se guardaron en una lista de Power Bi para ser mostradas por informes, pero que se quiera pasar a Excel. Con esto ya se puede generar una gran variedad de informes que guardar en Excel, como listas de clientes, o en PDF, como informes de las ventas del día.

Una vez obtenidos los datos hay que mantenerlos actualizados para mostrar los correctos cuando se generen los informes desde ellos. Las consultas se pueden configurar para que se actualicen al abrir la hoja de cálculo, pero para actualizar automáticamente se utilizaron scripts de Powershell para automatizar la actualización y la generación de los informes.

```
$app = New-Object -comobject Excel.Application
$app.Visible = $True
$app.DisplayAlerts = $FALSE
$wb = $app.Workbooks.Open("Excel a abrir")
$wb.Name
$wb.RefreshAll()
Start-Sleep -s 10
```

```
$wb.Save()
$wb.Saved = $true
$wb.Close()
$app.Quit()
```

Este script de Prowershell abre la hoja de cálculo, la actualiza y la cierra. Hay que tener en cuenta que la conexión OLAP se actualiza antes que las conexiones de tablas, por lo que, si se hace referencia con una tabla a los datos de una consulta, no hará falta actualizar 2 veces.

Powershell funciona con objetos, así que se empieza creando el objeto de Excel para poder abrir la hoja de cálculo, si tenemos que cargar datos de Power Bi con el Excel la función de abrir en segundo plano no actualiza esos datos. Entre el comando de actualizar, Refreshall(), y el de guardar la hoja de cálculo, Save(), se añadió una pausa de 10 segundos para dejar tiempo al Contenedor de Mashup de Microsoft a terminar de procesar la conexión de datos y así no tener problemas al guardar.

Para un montaje que requiera mostrar listas de datos, como por ejemplo una lista de clientes para cada uno de los representantes, se generaron varias tablas con los datos de clientes filtrados para cada representante. Además, se llamó a cada página de la hoja de cálculo como el representante filtrado, así con este código podemos dividir la hoja de cálculo en varias para los representantes:

```
foreach($Worksheet in $wb.Worksheets) {
  $ Worksheet.Connections.Item(1).Delete()
if($Worksheet.Name -EQ "REPRESENTANTE1"){
        $pp = "E:\DOCS\VENTAS\INFORME REPRESENTANTE1 "+(Get-
        Date).year+".xlsx"

        Set-ItemProperty -path $pp -name IsReadOnly $false

        $wb.SaveAs("E:\DOCS\VENTAS\INFORME REPRESENTANTE1
        "+(Get-Date).year+".xlsx"

        Set-ItemProperty -path $pp -name IsReadOnly $true
}
```

Este código barre todas las paginas existentes en la hoja de cálculos, donde, en cada una, elimina las conexiones existentes, para evitar que cuando un usuario abra la hoja de cálculo se intente actualicen los datos y les salga un error si no tuviesen permisos de acceso. Luego comprueba el nombre de la página, si se llama "REPRESENTANTE1", en este caso, realizará una serie de pasos para poder guardar la página como un Excel en la carpeta adecuada, dando acceso al representante solo a su información.

Primero hay que tener en cuenta que este es un proceso reiterado, actualizando diariamente la hoja de cálculo para proporcionar datos actualizados. En este caso el archivo resultante se pidió que fuese de solo lectura para que los representantes no pudiesen modificarlo, así que cada vez que se quiera actualizar es necesario eliminar el estado de solo lectura, con el código "Set-ItemProperty -path $pp -name IsReadOnly $false", para después sobrescribirlo y devolverle el estado de solo lectura.

Para generar archivos PDF de los informes, hay que tener en cuenta el tamaño de lo que se quiere convertir, ya que si no está bien ajustado el resultado no será el deseado.

```
$xlFixedFormat = "Microsoft.Office.Interop.Excel.xlFixedFormatType" -as [type]

foreach($Worksheet in $wb.Worksheets) {

        if($Worksheet.Name -EQ "REPRESENTANTE1"){

        $TARGETDIR = "E:\DOCS\VENTAS"

        $Worksheet.
        .Range("A1:M33").ExportAsFixedFormat($xlFixedFormat::xlTypePDF,
        $TARGETDIR+"\ALBARANES.PDF")

    }
```

El Comando ExportAsFixedFormat es el método más sencillo de usar para guardar hojas de cálculos en otros formatos. Este comando necesita que se le pase el formato al que tiene que guardar el archivo, que puede ser en formato PDF o Microsoft XPS, para ello requiere que se llame la función xlFixedFormatType

El código para guardar a PDF se puede invocar sin especificar el tamaño fijo del Excel que se pasa al PDF, omitiendo la parte de .Range("A1:M33"). del código. Si se delimitan correctamente los bordes del Excel se puede crear un proceso que genera PDF que pueden crecer o menguar en tamaño dependiendo del tamaño de la hoja a convertir.

Este montaje está diseñado para, también, guardar los archivos en carpetas anuales, por lo que a todos los Scripts se les añadieron comandos para guardar en las carpetas del año actual.

```
if(!(Test-Path -Path $TARGETDIR)){
        New-Item -ItemType directory -Path $TARGETDIR
}
```

Este código comprueba si existe una carpeta con el nombre de la variable $TARGETDIR, esta variable es el nombre de la carpeta, creándola si no existe. A la variable se le añadió el comando (Get-Date).year para que el nombre a buscar contenga el año actual, creándose así una carpeta nueva cada año de forma automática y guardándose en esta los archivos de este año.

Hasta ahora todos estos procesos guardan los archivos en carpetas compartidas en el servidor que ya ese era el montaje anterior cuando el proceso era manual. Pero también es posible, mediante Powershell, enviar los archivos generados por correo:

```
$EmailTo = "Correo remitente"
$EmailFrom = "Correo receptor"
$Subject = "ALBARANES Y PRESUPUESTOS DEL DIA "+((Get-Date).day-1)+"/"+(Get-Date).month+"/"+(Get-Date).year+" (no responder)"
$Body = "Mensaje automático con la lista de albaranes emitidos y presupuestos aceptados del día."
 $SMTPServer = "smtp.office365.com"
$filenameAndPath="Archivo a adjuntar 1"
$filenameAndPath2="Archivo a adjuntar 2"
$SMTPMessage = New-Object System.Net.Mail.MailMessage($EmailFrom,$EmailTo,$Subject,$Body)
$attachment = New-Object System.Net.Mail.Attachment($filenameAndPath)
$SMTPMessage.Attachments.Add($attachment)
$SMTPClient = New-Object Net.Mail.SMTPClient($SMTPServer, 587)
$SMTPClient.EnableSsl = $true
$SMTPClient.Credentials = New-Object System.Net.NetworkCredential("Correo remitente","Contraseña del correo remitente");
```

```
$SMTPClient.Send($SMTPMessage)
```

Este código utiliza el servidor API de SMTP de Microsoft 365 para enviar el correo sin necesidad de utilizar Outlook en el servidor. Este código, sin embargo, no encripta la contraseña del usuario, para ello habría que usar:

```
$pass = Get-Content "C:\Folder\Password.txt" | ConvertTo-SecureString -AsPlainText -Force
```

Y, por último, queda programar que todas las actualizaciones se realicen periódicamente. Para ello se utilizó el programador de tareas de Windows, que permite programar las tareas con la periodicidad deseada además de poder ver si fallo el proceso, como se ejemplifica en la figura 6.9.

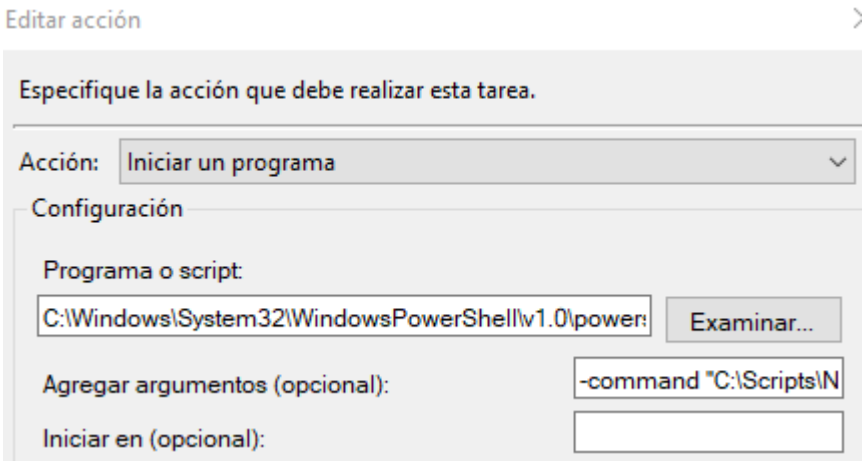

Figura 6.9- Programador de tareas de Powershell.

En la figura anterior se programó para ejecutar el programa de Powershell directamente, pero este proceso resultaba en demasiados errores, por lo que, como alternativa, se añadió un paso intermedio con un .bat. El .bat contenía las órdenes para ejecutar los scripts de Powershell que se querían ejecutar y en el programador de tareas se indicaba que se ejecutase únicamente este archivo .bat.

```
@ECHO OFF
PowerShell.exe -Command "Dirección del Powershell1"
PowerShell.exe -Command "Dirección del Powershell2"
```

Este método resultaba muy útil en el sentido de que se pueden agrupar los scripts que se actualizan con la misma periodicidad en la misma tarea programada, en el orden deseado. Además, te aseguras de que hasta que un proceso no termine uno de ellos el siguiente no se ejecuta, esto resulta extremadamente útil si tienes Excel que dependan de otros, y para reducir la carga del servidor evitando que cargue a la vez 2 Excel que tengan consultas con grandes volúmenes de datos.

6.3.- Tratamiento de correos

En empresas que manejan interacciones con otras empresas o clientes, es normal tener un gran flujo de correos haciendo referencia a códigos de procesos, como por ejemplo presupuestos, ordenes, facturas y recibos. Normalmente estos correos los recibe una persona que, consultando la base de datos, redirige los correos a los trabajadores encargados de ese proceso, ya sea por a que sucursal hace referencia o qué tipo de código incluye el mensaje de los tipos que manejan.

La persona que reenvía estos correos no trata el problema del correo, únicamente los reenvía a la persona que lo va a tratar. Es por esta razón que un sistema solido de tratado de correo es muy importante en estos tipos de empresa.

Este flujo se encarga de eso exactamente, cada vez que se reciba un correo en un buzón determinado se realiza una serie de procesos acabando en el reenvió del correo a la cuenta de correo correcta o, en caso de no encontrar nada, enviándolo a un buzón donde una persona real se encargue de tratarlo. Esta última situación surgiría en situaciones extremas, donde no se encontró ninguna coincidencia en el correo, bien sea por algún error en el remitente o por una situación especial que podría solucionarse personalizando más el filtrado para estos casos especiales. Reduciéndose, gracias a esto, el trabajo que una persona física tenga que realizar para reenviar los correos a los buzones correctos.

Este flujo desarrollado en el trabajo es una versión antigua del que está funcionando actualmente en la empresa. Debido a que esa versión contiene una gran cantidad de personalizaciones para solucionar los reenvíos al buzón de tratado que se comentó anteriormente.

El proceso está dividido principalmente en tres partes: la detección de los posibles códigos, la búsqueda de coincidencias en la base de datos y la detección de coincidencias en los archivos adjuntos del correo. La detección de coincidencias en los adjuntos se separó ya que para el cliente era prioritario las coincidencias en el contenido del correo.

6.3.1.- Detección de posibles coincidencias

La primera parte consiste en encontrar en el asunto y en el cuerpo todas aquellas cadenas que puedan ser códigos de la base de datos. Este paso depende el tipo de información que se maneje, en este caso los códigos a detectar pueden contener números, letras y caracteres, lo cual complica el montaje aumentado los posibles códigos.

Esta primera parte es muy personalizable, ya que cuanto más se conozca de las características de los códigos mejor se puede reducir la lista de posibilidades y más rápido será el proceso de búsqueda.

Para este caso se sabe que, aunque existen también caracteres que no son números, hay como mínimo 5 números seguidos en el código. Este simple detalle permite reducir enormemente la lista de posibles códigos, acelerando el procesado de cada correo.

Al final el primer bloque quedo como se ve en la figura 6.10:

Figura 6.10- Bloque 1 del flujo de correos.

El funcionamiento de este bloque es el siguiente:

- El primer bloque representa el desencadenante que marca el inicio del proceso: cuando se recibe un correo.
- **Html to text**: El desencadenante recibe el cuerpo del correo electrónico en formato html, para evitar errores durante la detección de códigos se pasa a formato texto.
- **Inicializar variables entrada asunto**: Este bloque recoge el asunto del correo y se pasa a mayúsculas. Para evitar errores de comparación con los códigos de la base de datos, ya que en este caso todos los códigos están en mayúsculas.

```
toUpper(triggerBody()?['subject'])
```

- **Inicializar variable ARRAY QUITAR:** Esta variable consiste en la lista de caracteres que se eliminarán del asunto y del cuerpo. Si los códigos solo pudiesen contener números aquí se podrían añadir todos los caracteres del alfabeto para que quedasen solo los números.
- Los siguientes dos bloques se encargan de pasar a través de cada variable del array de quitar y eliminarlos del asunto. Este proceso se reproduce con el cuerpo del correo.

  replace(variables('ENTRADA ASUNTO'),item(),' ')
- **NUMERO REENVIOS:** Esta variable marca el número de veces que se ha detectado una coincidencia y se ha reenviado el correo tratado. Esta parte es importante ya que, en el cuerpo del correo, y en los archivos adjuntos, se quiere saber si se encontró más de una coincidencia, por políticas de la empresa.
- **CADENAS DE NÚMEROS ASUNTO Y CUERPO:** Estos dos bloques cogen la cadena después de habérsele eliminado los caracteres deseados y se divide en una serie de matrices separando por los espacios. De esta manera si es texto es, por ejemplo ‘Código 1234567’, este bloque acabara con 2 elementos: Código y 1234567.

  split(uriComponentToString(replace(uriComponent(variables('ENTRADA ASUNTO')), '%0A', ' ')), ' ')

  Además, los retornos de carro dejan \r\n en la cadena de texto, para eliminarlo del asunto y del cuerpo se pasa a formato URI, se reemplaza el código %0A, correspondiente al retorno de carro, y se devuelve al formato original
- **FILTRADO DE ASUNTO Y CUERPO:** Por último, en cuatro bloques, dos para asuntos y dos para el cuerpo, se filtran solo aquellos que, una vez eliminados todos los caracteres que sean números, contienen al menos 5 números seguidos.

```
{
    "inputs": {
        "from": "@variables('CADENA NUMEROS ASUNTOS')",
        "where": "@greaterOrEquals(length(replace(replace(replace(replace
(replace(replace(replace(replace(replace(replace(replace(replace(replace(
replace(replace(replace(replace(replace(replace(replace(replace(replace(r
eplace(replace(replace(replace(replace(replace(replace(replace(replace(re
place(replace(replace(replace(replace(replace(replace(replace(item(), 'A'
, ''), 'B', ''), 'C', ''), 'D', ''), 'E', ''), 'F', ''), 'G', ''), 'H', '
'), 'I', ''), 'J', ''), 'K', ''), 'L', ''), 'M', ''), 'N', ''), 'Ñ', ''),
 'O', ''), 'P', ''), 'Q', ''), 'R', ''), 'S', ''), 'T', ''), 'U', ''), 'V
', ''), 'W', ''), 'X', ''), 'Y', ''), 'Z', ''), '-
', ''), '_', ''), '.', ''), ',', ''), '/', ''), '\\', ''), '%', ''), '&',
 ''), '[', ''), ']', ''), '(', ''), ')', ''')), 5)"
    }
}
```

El segundo bloque se asegura que el código resultante contiene otros caracteres que no sean 0.

6.3.2.- Comparación con la base de datos

La segunda parte del flujo se encarga de recoger la lista de posibles coincidencias y pasarlas por la base de datos para comprobar si hay alguna coincidencia exacta y, si la hay, reenviar el correo a una cuenta de correo tratado. Aquí sería posible aplicar un sistema adicional para reenviarlo a diversos correos, dependiendo de las variables obtenibles con las características de la coincidencia.

En este montaje se creó una prioridad de detección, donde si se detecta una coincidencia del "Código1" no se pasa a comprobar si existen coincidencias del "Código2" y si no se encuentran coincidencias de "Código1" ni de "Código2" es cuando se pasa a buscar coincidencias de "Código3", quedando un sistema como el de la figura 6.11.

Además, para evitar perder recursos buscado coincidencias cuando ya se ha reenviado el correo tras encontrar una coincidencia, se usa la variable generada anteriormente "NUMERO REENVIADO". Para ello se usa como un booleano, comprobando si es 0 para proceder y si es 1 se salta el proceso, como se ve en la figura 6.13 para el asunto y en la 6.14 para el cuerpo.

En cuanto al cuerpo, como ya se explicó, se tendrá que seguir haciendo comprobaciones, aunque se encuentre una coincidencia. Para ello, se usa otra variable llamada "NUMERO COINCIDENCIAS CUERPO" la cual, tras comprobar toda la lista de posibles códigos, se comprobará su valor: si hubo más de 1 coincidencia se manda un correo avisando de este hecho, si solo hay una coincidencia, se aumenta la variable "NUMERO REENVIOS" y se reenvía el correo a la cuenta de tratados y si no hubo coincidencias se pasara al siguiente tipo de Código, con un conmutador como se ve en la figura 6.14.

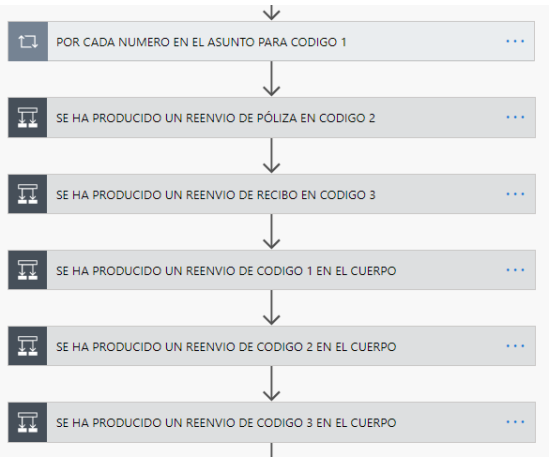

Figura 6.11- Bloque 2 del flujo de correos.

Todos los bucles de la figura 6.11 tienen la misma estructura, pasan por cada valor de la matriz "CADENA DE NUMEROS" y comprobar si existe en la base de datos, como se ejemplifica en la figura 6.12. Entre ellos solo cambia la tabla de la base de datos a la que hace referencia para cada código:

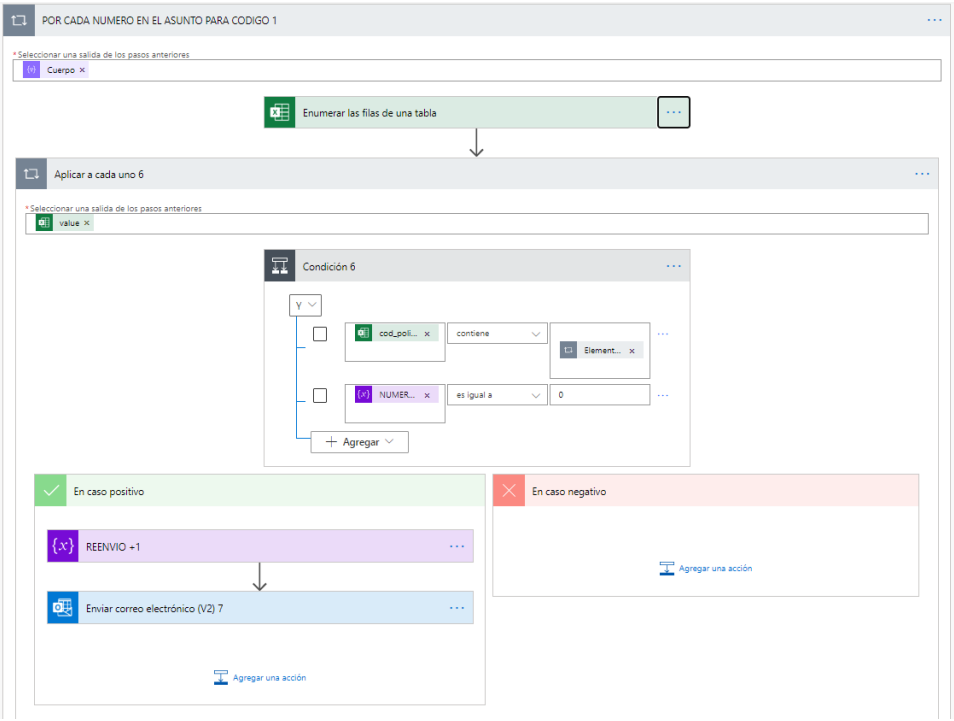

Figura 6.12- Bucle de comprobación de coincidencias.

Para este ejemplo, se usó una tabla en una hoja de cálculo de Excel para sustituir a las tablas de una base de datos.

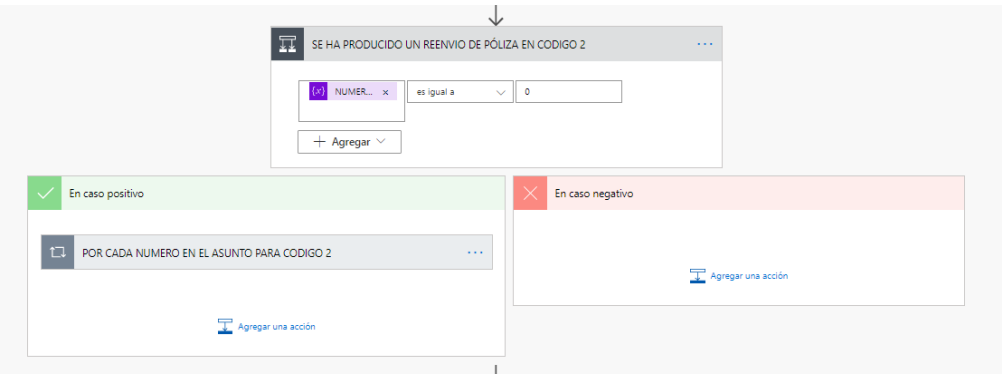

Figura 6.13- Interior del bloque condicional del asunto.

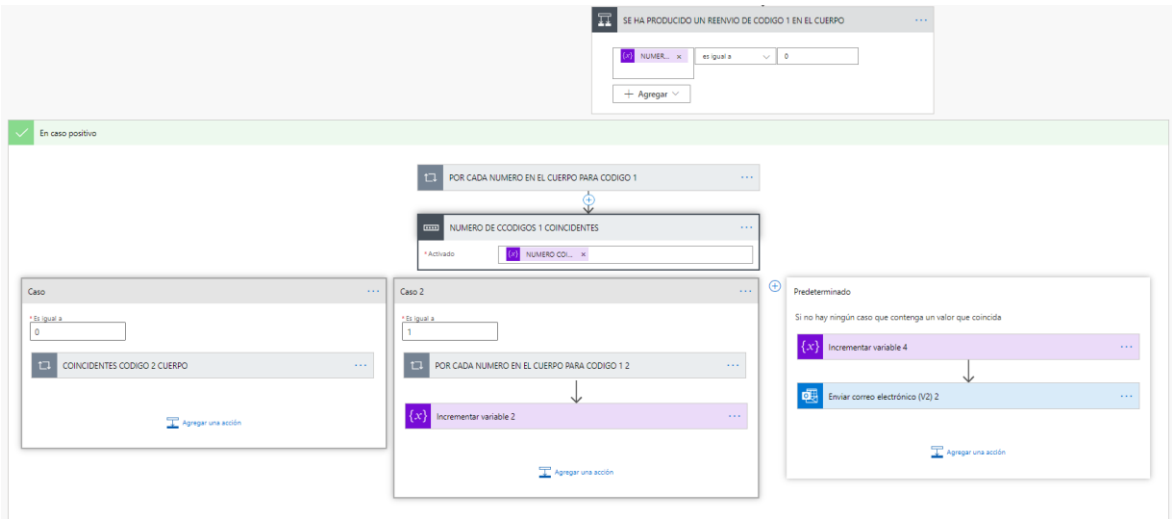

Figura 6.14- Interior del bloque condicional del cuerpo.

6.3.3.- Archivos adjuntos

Si no se ha encontrado ninguna coincidencia ni en el asunto ni en el cuerpo del mensaje se pasa a comprobar los archivos adjuntos. Por petición del cliente solo se tratan archivos adjuntos en formato PDF y solo si hay 1 solo, por eso el proceso empieza comprobando el número de archivos PDF para decidir si realizar o no esta parte de la comprobación.

Como el archivo es de formato PDF se necesita realizar una conversión a un formato de texto que el programa entienda. Sin embargo, actualmente Power Automate no cuenta con dicha opción, así que es necesario contratar un servicio de una empresa tercera, como Cloudmersive en este ejemplo, para realizar esa función. Los precios entre empresas varían dependiendo del número de iteraciones en paralelo que quieras y el número máximo de usos al mes.

Tras obtener el contenido del PDF en formato texto se realiza el mismo procedimiento que en el cuerpo del correo, como se ve en la figura 6.15.

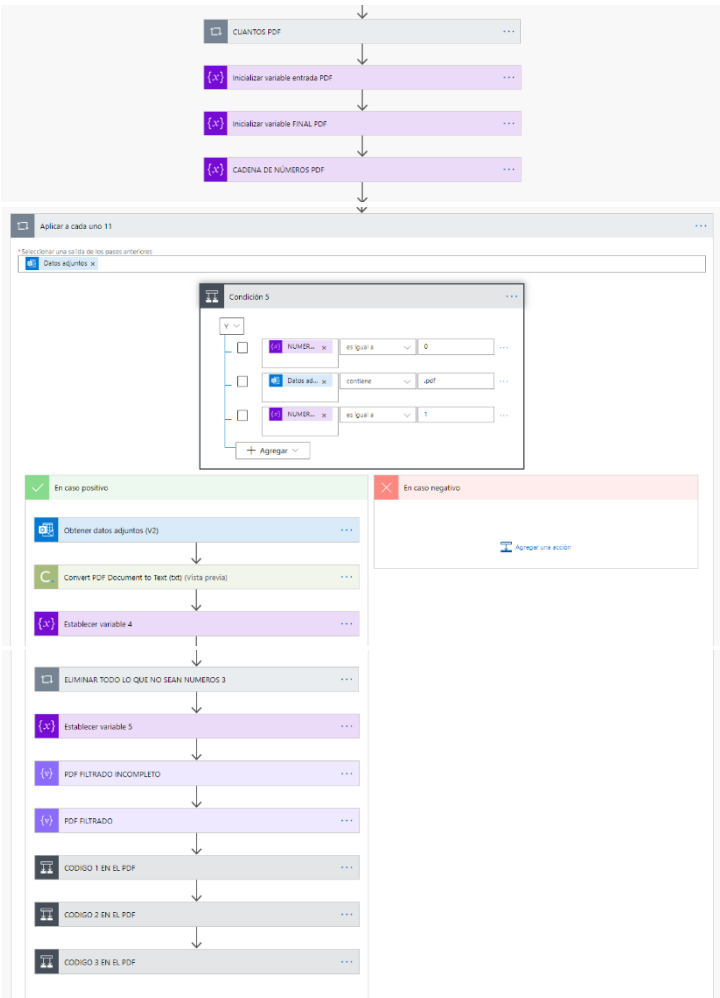

Figura 6.15- Bloque 3 del flujo de correos.

Si se llega al final del flujo y el número de reenvíos sigue siendo 0 significa que no se ha encontrado ninguna coincidencia y puede necesitar ser analizados manualmente, bien por problemas en la detección o porque no es un correo que contenga ningún código de la base de datos.

6.4.- Presupuesto aproximado

En este apartado se reflejará el presupuesto aproximado del apartado de automatización para poder montar todo lo explicado. Se separarán entre aquellas que se necesitase una por usuario y las que no.

LICENCIA	PRECIO
Microsoft 365 E3	19,70 €/mes/usuario
Power Bi Pro	8 €/mes/usuario
Power Automate Premium	13 €/mes
Cloudmersive	17,40 €/mes
Precio final	30,40 €/mes + 27,70 €/mes/usuario

Tabla 6.1 – Tabla de precio estimado mínimo.

El precio con €/mes/usuario tiene que ser como mínimo de 1 usuario. Pero la combinación de licencias de Microsoft 365 y de Power Bi depende del montaje final deseado, y no tiene por qué ser de 1 a 1.

7.- PROBLEMAS

Al realizar los proyectos de automatización surgieron varios problemas, algunos más graves que otros, que ralentizaron e incluso cambiaron por completo el enfoque que se tenía sobre cómo realizarlo. Entre ellos, el más importante de todos es que al utilizar herramientas proporcionadas por Microsoft, que, aunque aceleran y facilitan la realización del proyecto, te encuentras limitado a las opciones de configuración de dicha herramienta. Si necesitabas más de ella te ves forzado a sacrificar algo a cambio, ya sea velocidad del proceso o la sencillez que podría haber tenido, o a tener que programar una extensión tú mismo si no lo ha hecho la comunidad.

Un ejemplo muy importante de este hecho es al usar Power Automate, al ser un programa basado en bloques si no existe un bloque o una función que haga lo que necesitas te ves obligado a improvisar, usando una herramienta distinta o un programa de terceros. Con este problema hay muchas posibilidades: que exista un conector de otra empresa asociada con Microsoft que te proporcione lo que necesites, como el caso de realizar un OCR a un PDF, o tienes que renunciar a esa idea y realizarlo de otra forma.

Además, te encuentras atado a las políticas y cambios de Microsoft, pues ellos tienen la última palabra a la hora de tomar las decisiones.

Durante uno de los proyectos se necesitaba accederá la base de datos situada en un servidor de una empresa externa al cliente, como no podíamos levantar la puerta de enlace hacia ese servidor, aunque existía un conector en Power Automate para dicha empresa, resultó imposible acceder a los datos. Por lo que se tuvo que usar Excel de intermediario, obteniendo los datos usando ODBC, para poder hacer que los datos llegasen a Power Automate. Convirtiendo un montaje que podría haber sido independiente del hardware del cliente en otro montaje que necesita Excel y Powershell corriendo en el servidor y, aunque no es un montaje muy problemático, aumenta el número de posibilidades de producirse un error.

Respecto a usar Excel como intermediario para recoger los datos, aunque es una herramienta muy útil para muchas situaciones, presentó a lo largo del desarrollo de los proyectos una serie de problemas.

- La actualización en segundo plano no funciona con datos obtenidos de un servidor SQL. Lo que obligaba a abrir físicamente el Excel para poder actualizar la consulta con la base de datos, haciendo necesaria una cuenta específica para el proceso para que trabajadores no puedan interferir en la actualización ni les paren el trabajo. Esto también impedía que el programador de tareas ejecute las tareas en segundo plano, obligando a dicha cuenta a estar

al menos en suspensión, es decir que al reiniciar el servidor el proceso se podría congelar si no se prepara el programador de tareas.

- El hecho de tener que abrir el Excel con script significa que los códigos VBA programados para iniciar al arrancar el Excel no se lanzaban (este problema tampoco es muy importante ya que se pueden ejecutar las funciones igualmente desde el propio script que lo abre).
- Al tener que abrir, guardar y cerrar el Excel con una conexión a base de datos, el programa podría intentar guardar el Excel antes de que el proceso "Microsoft.Mashup" lo libere, produciéndose un problema de "permisos", como se muestran en la figura 7.1. Este error evita que el Excel se guarde, parando el proceso y congelándolo hasta que haya intervención humana (o se obligue a cerrarlo con algún código de contención).

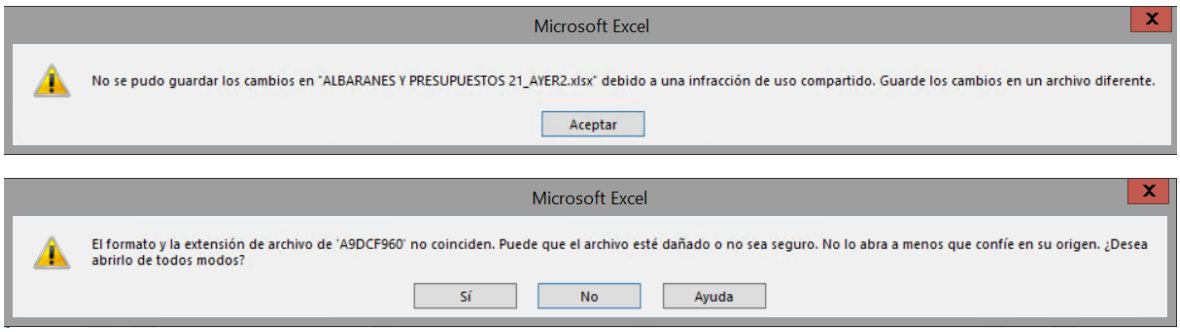

Figura 7.1- Error porque el Mashup no termino la conexión con el Excel (arriba) y Excel intentando guardarlo con otro nombre (abajo).

Este error, además de significar también que puedes no tener permisos sobre el Excel, es el mismo error que si intentas guardar un Excel que tiene otra persona abierta (problema que suele pasar al trabajar en carpetas compartidas). Lo que quiere decir que el proceso "Mashup" no libera el proceso del todo cuando se termina de actualizar una conexión OLAP y produce este error.

Para arreglar este problema fue necesario modificar el código, guardándolo con un nombre distinto, como por ejemplo "exceltrabajado_backup", para luego abrirlo y devolverlo al original, alargando el tiempo del proceso (pero no mucho si evitas actualizar las conexiones cuando abres el segundo Excel):

```
…
$wb.SaveAs("E:\DOCS \ exceltrabajado_backup.xlsx")
…
$wb.Close()
$app.Quit()
[System.Runtime.InteropServices.Marshal]::ReleaseComObject($app)
#Y tras cerrar el Excel de trabajo abrimos el nuevo creado.
$app = New-Object -comobject Excel.Application
```

```
$app.Visible = $True
$app.DisplayAlerts = $FALSE
$wb=$app.Workbooks.Open("E:\DOCS \exceltrabajo_backup.xlsx")
$wb.Name
$Worksheet = $wb.Worksheets.item(1)
$wb.SaveAs("E:\DOCS \exceltrabajo.xlsx")
$wb.Saved = $true
$wb.Close()
$app.Quit()
[System.Runtime.InteropServices.Marshal]::ReleaseComObject($app)
```

- Si se produce algún error que congele un proceso, si hay más de un sistema similar corriendo en paralelo que se alimente de una base de datos online (como Power Bi si se sigue el montaje explicado anteriormente), este podría encontrarse con un problema al conectarse con la base de datos y pedir reintroducir la cuenta de Microsoft, produciéndose una caída en cadena del proceso.

Power Bi, aunque no presento muchos problemas a la hora de su uso, al ser sencillo de manejar a bajo nivel, el número limitado de módulos a tu disposición y su baja personalización llevó a producir dificultades y a tener que buscar alternativas mucho más complicadas de lo normal (como los métodos explicados en el punto 6.1 para mostrar información adicional de un código). Además, el hecho de necesitar Power Bi Pro por cada usuario que quiera acceder al informe con un enlace o iframe, a no ser que se haga público, limita mucho su uso en empresas con muchos trabajadores pero que no quieran gastarse más al mes por usuario.

Por último, aunque el script de enviar correo correos electrónicos con los informes adjuntos funcionó durante 2 meses, empezó a fallar hasta que finalmente dejaron de enviarse los correos, así que se decidió cambiar al método de envío directo. Que consiste en utilizar el punto final MX del dominio contratado con Microsoft para sustituir al servidor SMTP de office 365 para enviar el correo, como explica Microsoft en la figura 7.2.

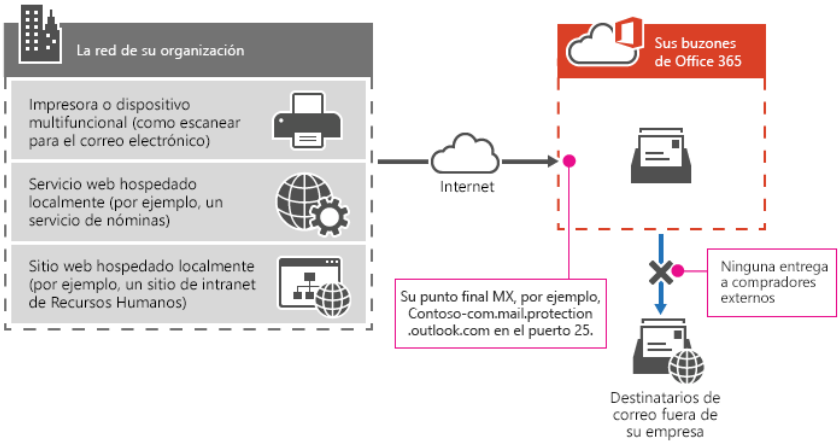

Figura 7.2- Método de envío directo explicado por Microsoft.

Si embargo este método devolvía un mensaje de error de que la IP estaba bloqueada, pero se comprobó que ese no podía ser el caso, por lo que se cambió a un script que usaba la cuenta ya configurada de Outlook para enviar el correo gracias a que Powershell tiene objetos para todas las herramientas de Microsoft.

```
#Se crear el COM de Outlook
$Outlook = New-Object -ComObject Outlook.Application
#Se crea un Outlook MailItem llamado Mail usando CreateItem()
$Mail = $Outlook.CreateItem(0)
$Mail.To = "Receptor del correo"
$Mail.Subject = "Asunto"
$Mail.Body = "Cuerpo"
#Se envía el mensaje
$Mail.Send()
$Outlook.Quit()
[System.Runtime.Interopservices.Marshal]::ReleaseComObject($Outlook) | Out-
Null
```

Pero este método funcionó solo un mes. El .pst de la cuenta en el servidor empezó a corromperse, debido a que los correos estaban siendo mandados de forma forzada y no se estaban registrando correctamente en el archivo .pst, pero si en la nube, produciéndose un problema de sincronización que detenía las nuevas iteraciones del script que trataban de conectarse a la aplicación de Outlook.

Finalmente se utilizó Power Automate para que todos los días a la misma hora coger los archivos del servidor, con una petición GET, y mandarlos por correo usando el propio Power Automate, con un montaje sencillo como se ve en la figura 7.3.

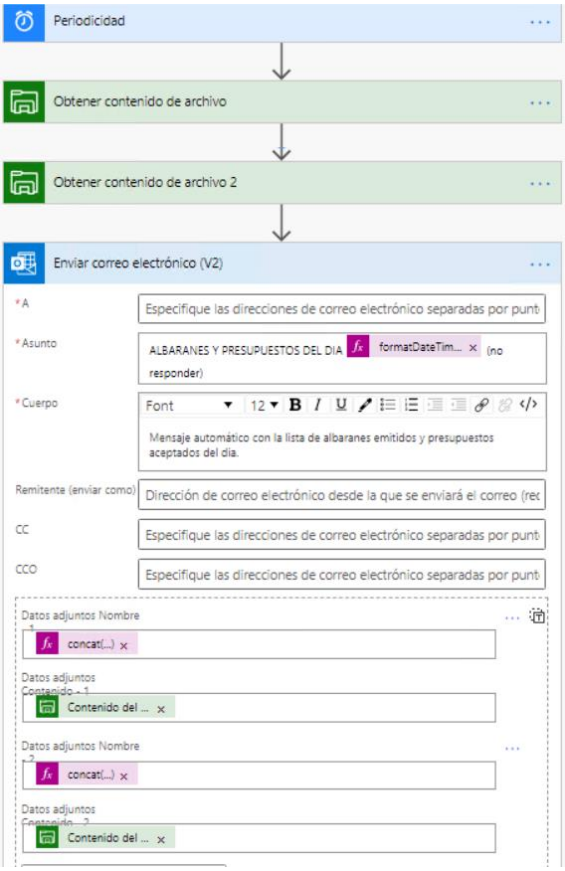

Figura 7.3- Solución final para los correos diarios.

Como ya había una puerta de enlace instalada para el resto de los procesos esta nueva función no trajo ningún nuevo gasto al proceso ya montado, y te asegura que los correos se vayan a mandar todos los días a esa hora exacta.

Pero esto también es su problema, con el montaje anterior el script para mandar el correo estaba situado después de generar los archivos a enviar, eso aseguraba que el correo solo se fuese a mandar cuando los archivos estaban correctamente generados. Ahora, si el proceso se ralentizase por alguna razón, se podría mandar por error un archivo no actualizado. Lo que hace necesario añadir un proceso adicional que compruebe la última fecha de actualización y esperar a que este correctamente actualizada, enviando un correo de aviso si tardase demasiado.

8.- MEJORAS

Los desarrollos mostrados en este trabajo provienen de la experiencia que he obtenido trabajando solo en proyectos de este ámbito para distintas empresas con distintas situaciones, desde el tipo de servidores SQL que usaban hasta el tipo de salidas de información que esperaban.

Por lo tanto, hay montajes que se realizaron, en su momento, creyendo que eran la mejor solución, pero de los que surgieron problemas con el tiempo. Algunos hasta meses después de ponerse en marcha que, aunque no resultaban en problemas visibles ni incorregibles, demostraron que había mucho en lo que se podía mejorar.

Aunque la solución de utilizar Power Bi como intermediario entre Excel y la base de datos daba muy buenos resultados debido a la rapidez y flexibilidad del Query de Power Bi, el montaje de la conexión OLAP por parte de Excel ralentizaba mucho el proceso final. Esto es debido a que se hacia una consulta a la información de Power Bi, no a la de SQL, conectándose a todos los datos sin posibilidad de filtrarlos como con la consulta a SQL.

En cuanto a los scripts de Powershell tienen el problema que, como están ahora mismo, si se produjese un error no se sabría nada hasta que alguien se diese cuenta en los datos del resultado final o si alguien entrase a comprobar el estado del servidor. Lo cual es un fallo de diseño que puede traer muchos problemas en el futuro. Para solucionar esto, Powershell tiene unos comandos llamados try/catch:

```
Try {

        #Comandos a realizar

}
Catch {

        #Que hacer cuando hay un error

}
```

El primer bloque dentro del Try seria el mismo código usado hasta ahora que realizaría la misma función de antes, si no se produce ningún problema el proceso se terminará sin diferencia. Pero en el momento en el que se produzca un mensaje de error que saldría en el interfaz de Powershell, se pasará a segundo bloque donde se tratará el proceso en caso de error.

Con esto podemos realizar maniobras de control estándar para cualquier tipo de error, como: cerrar todas las estancias de Excel, de Mashup y enviar un correo avisando del error que se ha producido.

Además, este comando permite poner varios 'Catch' seguidos. De forma que si poner 'catch [ERROR.ESPECIFICO]' solo se entrara en ese 'Catch' si se cumple el tipo de error marcado entre los corchetes, por lo que si sabemos el tipo de error podemos actuar adecuadamente y no necesitar enviar un correo de aviso si no es un error que lo requiera.

Una cosa que tener en cuenta con este comando es que solo se pasa a los catch los errores 'terminales'. Así que errores que Microsoft no considere terminales no se trataran, una manera de solucionarlo es añadir -ErrorAction Stop después de los comandos que puedan producir error, esto hará que si se produjese un error en la línea del script que contiene este comando se saltaría al 'Catch' que no tenga ningún error especificado.

Powershell también cuenta con módulos instalables que le permiten acceder a archivos de Sharepoint y Onedrive desde la web, sin necesidad de instalarlo en el servidor o de configurar la cuenta. Esto podría aliviar la carga de datos del servidor realizando el Query desde Power Automate y después accediendo a los datos desde Powershell y, con una programación más avanzada orientada a modificar el contenido del Excel, transformar los datos y crear los informes sin tener que lanzar una conexión OLAP.

En cuanto a Power Bi, aparte de construir los informes de una manera más eficiente, existe una opción mucho más avanzada pero que no se tuvo en cuenta en el trabajo, ya que estos proyectos estaban orientados a pequeñas o medianas empresas que querían emprender el primer paso a digitalizar su trabajo y el precio de esta opción se disparaba. Pero, con los avances de Microsoft con Azure y sus recientes reducciones de precios, se puede considerar que esta opción pronto dejara de estar fuera del alcance de muchos: Power Bi Embedded, el cual Microsoft representa con la imagen 8.1.

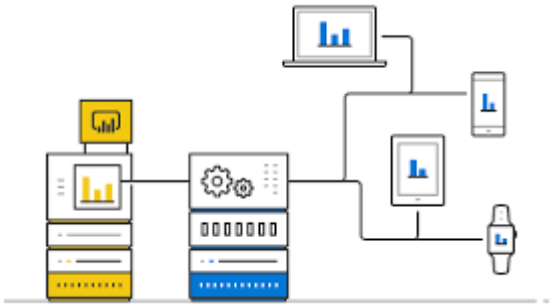

Figura 8.1- Análisis de Microsoft de Power Bi Embedded.

En resumen, con Power Bi Embedded Microsoft te proporciona las bases para crear tu propia aplicación, como el de la figura 8.2, alojamiento en Azure y software fácilmente programable, en los que puedes introducir tus informes como los de Power Bi estándar de una manera más interactiva para el usuario.

Figura 8.2- Imagen de ejemplo de Microsoft de Power Bi Embedded.

Este montaje requiere solo una licencia de Power Bi Premium que permite a un número ilimitado de miembros usar la aplicación, además como puedes crear informes que mostrar distintos para usuarios. Además, puedes crear una versión para clientes y otra para tus empleados, por ejemplo, mostrando las ventas totales de tu empresa a tus trabajadores y en la misma aplicación cada cliente pueda inspeccionar el estado de su pedido.

En cuanto al futuro de la automatización, consiste en el uso de algoritmos de aprendizaje e inteligencias artificiales para poder adaptarse de forma activa a los cambios en el flujo de datos:

"A key benefit of Deep Learning is the analysis and learning of massive amounts of unsupervised data, making it a valuable tool for Big Data Analytics where raw data is largely unlabeled and un-categorized." [14]

El trabajo principal de las máquinas de aprendizaje automáticas es el de estudiar los patrones existentes en los datos de entrada actuales para predecir lo que podría suceder con los datos que están por llegar. La veracidad de los datos usados para la máquina de aprendizaje es muy importante, ya que si se un conjunto de datos que no representen correctamente la situación actual podría llevar a la maquina a tomar decisiones incorrectas por muy avanzada que sea la máquina, mientras que cuanta mejor sea la presentación mayor rendimiento se obtendrá.

9.- CONCLUSIONES

En este proyecto se ha hablado de la importancia de manejar la información del mercado, pues quien analice mejor los datos obtenidos podrá tomar las decisiones más optimas y controlar el mercado. Pero para poder realizar algoritmos de aprendizaje y poder prever lo que va a suceder en el futuro, es esencial, en primer lugar, aprovechar la información generada, un hecho que muchas empresas ni siquiera realizan. Es por eso por lo que el enfoque de este trabajo ha sido el de realizar el primer paso hacia este futuro. Empezando por las representaciones simples de datos, en forma de automatizaciones de procesos, para ayudar a empresas a entender las grandes posibilidades que proporciona la digitalización de estos.

Para ello se han utilizaron, como base, únicamente herramientas de Microsoft, una compañía conocida por todos y con la que se tiene un buen nivel de confianza, como para que las empresas le permitan, en primer lugar, manejar la delicada información de sus bases de datos. Y, aunque se encontraron varios problemas durante el proceso, como se explicó en el proyecto, se llegaron a realizar, con satisfacción para la empresa, automatizaciones útiles, las cuales, además de evitar errores humanos que surgían cuando tenían que producir los informes a mano, les quitaban una enorme cantidad de trabajo esencial pero tedioso a las empresas. La sorpresa de algunas llegaba hasta el punto de, una vez se terminaba una automatización, organizar la digitalización de un proceso distinto, acercándose poco a poco a ese futuro que es la industria 4.0.

En cuanto a usar herramientas de Microsoft, aunque se encontró con diversos problemas, gracias a la posibilidad de contar con soporte rápido, por nuestra posición como Partner, y la extensa ayuda de la comunidad en los foros, nunca resultó en un montaje siendo imposible, solamente más complicado.

Y, aunque el proyecto pueda resultar simple, es esa simplicidad la que se quería mostrar, para que se entienda que el futuro del análisis de datos no está solo disponible para empresas multimillonarias, que el nivel básico puede ser alcanzado por cualquiera y que el siguiente paso no está tan lejos como parecía.

10.- BIBLIOGRAFIA

[1] LEE, Jay, et al. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia Cirp*, 2014, vol. 16, no 1, p. 3-8.

[2] STOCK, Tim; SELIGER, Günther. Opportunities of sustainable manufacturing in industry 4.0. *Procedia Cirp*, 2016, vol. 40, p. 536-541.

[3] HEEKS, Richard. Do information and communication technologies (ICTs) contribute to development?. *Journal of international development*, 2010, vol. 22, no 5, p. 625-640.

[4] EVANGELOS A, Kosmatos; NIKOLAOS D, Tselikas; ANTHONY C, Boucouvalas. Integrating RFIDs and smart objects into a UnifiedInternet of Things architecture. *Advances in Internet of Things*, 2011, vol. 2011.

[5] USTUNDAG, Alp; CEVIKCAN, Emre. *Industry 4.0: managing the digital transformation*. Springer, 2017.

[6] SHARMA, Shree Krishna; WANG, Xianbin. Live data analytics with collaborative edge and cloud processing in wireless IoT networks. *IEEE Access*, 2017, vol. 5, p. 4621-4635.

[7] GANDOMI, Amir; HAIDER, Murtaza. Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management*, 2015, vol. 35, no 2, p. 137-144.

[8] DING, Kai; JIANG, Pingyu. RFID-based production data analysis in an IoT-enabled smart job-shop. *IEEE/CAA Journal of Automatica Sinica*, 2017, vol. 5, no 1, p. 128-138.

[9] NICOLALDE, Fabián Constante, et al. Big Data analysis tools in IoT and their challenges in open researches. En *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, 2018. p. 1-6.

[10] THOMAS, Stephen. SSL and TLS essentials. *New Yourk*, 2000, vol. 3.

[11] BADIOZAMANY, Sobhan. Microsoft SQL server OLAP solution-A survey. 2010.

[12] EASTWICK, Michael W.; MISKIN, Gregory K. *Integration of legacy database management systems with ODBC-compliant application programs*. U.S. Patent No 6,240,417, 29 Mayo 2001.

[13] DARUWALLA, Feisal Y.; FORSTER, James R.; LITWACK, Mark W. *Technique and apparatus for using node ID as virtual private network (VPN) identifiers*. U.S. Patent No 6,693,878, 17 Feb. 2004.

[14] NAJAFABADI, Maryam M., et al. Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2015, vol. 2, no 1, p. 1.