Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN SISTEMAS DA INFORMACIÓN

Desenvolvemento do modelo analítico para a avaliación dos KPI dunha software factory

Estudante: José María Freire Ferreiro
Dirección: Tiago Manuel Fernández Caramés
Dirección: Manuel Ángel Linares Francós

A Coruña, setembro de 2019.

Aos meus pais e a Diana.

Agradecementos

En primeiro lugar, agradecer aos meus pais, miña irmá e ao resto da familia por todo o esforzo e apoio nos momentos máis difíciles.

A Aldaba por abrirme as portas para a realización deste traballo, e a todos os compañeiros, especialmente da área de BI.

Agradecer tamén aos meus compañeiros de piso e amigos que coñecín durante todos estos anos.

Aos meus titores Tiago Manuel Fernández Caramés e Manuel Ángel Linares Franco que me axudaron a levar a cabo este proxecto.

A Ana pola comprensión e apoio durante todo este tempo.

Resumo

Nos últimos anos, a implementación de Business Intelligence pasou de ser un 'luxo' de organizacións que podían permitirse destinar partidas de presupostos para tal efecto, a convertirse nunha necesidade en grandes e medianas empresas debido ao gran volume de datos que é preciso estruturar e analizar.

A principal motivación para a realización do presente Traballo de Fin de Grao é a necesidade de obter información de maneira visual e dende calquera dispositivo sobre todos os proxectos que se están a desenvolver na organización.

Para a visualización, os usuarios empregan a ferramenta de Microsoft Power BI. Os datos, previamente foron integrados nun modelo tabular creado en Analisys Services cuxa orixe é a ferramenta de xestión de proxectos na nube Azure Devops.

Para a obtención dunha parte dos datos foi preciso crear un servizo implementado en C# que obtén periódicamente os datos mediante API Rest de Azure Devops e os almacena nun ficheiro plano. O resto dos datos foi obtido directamente dende SSAS co protocolo oData.

Abstract

Over the last few years, the companies conception about Business Intelligence has changed a lot. From being an extra section, financed with leftover budget, to be something completely necessary, in part, because the big amount of data who need to be managed.

Considering this, companies have need for obtaining the data from all internally developed projects, and they need them structured, and accessible from any device; that's the main motivation of this Bachelor's Degree Dissertation.

In order to visualize this information, the employees use the tool Microsoft Power BI. The way followed to make these data available is divided into the next two steps. In a first step, data are obtained from the tool Azure Devops for project management and, in a second step, it is integrated in a tabular model that is created in Analisys Services.

It was necessary to create a service to obtain part of the data periodically, which, finally, are stored in a plain file. This service, developed in C#, uses an Azure Devops Rest API. The remaining data were obtained from SSAS, using oData protocol.

Palabras chave:

- Intelixencia de Negocio
- oData
- Power BI
- Visual Studio
- API Rest
- Azure Devops
- Visual Studio
- SSTD
- SSAS
- Modelo tabular

Keywords:

- Business Intelligence
- oData
- Power BI
- Visual Studio
- API Rest
- Azure Devops
- Visual Studio
- SSTD
- SSAS
- Tabular Model

Índice Xeral

Índice de Figuras

Índice de Táboas

Capítulo 1

Introdución

No primeiro capítulo faise unha pequena introdución a Business Intelligence, detállase o problema a resolver, a solución proposta e a viabilidade desta, e finalmente explícase a organización da memoria.

1.1 Introdución a Business Intelligence

A Intelixencia de Negocio (Business Intelligence) é unha das partes importantes na revolución tecnolóxica actual. Debido ao crecemento exponencial dos datos xerados, é preciso empregar sistemas que sexan capaces de analizar e crear información útil, e polo tanto, coñecemento.

Podemos definir entón o Business Intelligence como o conxunto de metodoloxías, aplicacións e tecnoloxías que permiten reunir, depurar e transformar datos, dende os sistemas transaccionais con información desestruturada, en información estruturada para a súa explotación directa (ou o seu análise) e conversión en coñecemento.

Os procesos ETL (extracción, transformación e carga) encárganse da tradución dun ou varios sistemas operacionais, normalizados e independentes a un único descentralizado, o cal contén os datos completamente integrados.

Para a extracción e análise de datos dende diferentes fontes emprégase o software BI. Con estas ferramentas, cabe a posibilidade de realizar análisis libres mediante asociacións de datos, gráficos... Este tipo de software permite as empresas realizar observación, comprensión, predición, colaboración entre departamentos e poder de decisión.

As persoas que precisan da Intelixencia de Negocio son na súa maior parte os que se atopan no nivel estratéxico, xa que estes son os encargados de poder tomar as decisións que afectarán a toda a organización.

1.2 Breve repaso á historia de Business Intelligence

A principios dos anos 70, nacen os sistemas de xestión de bases de datos e comezan a crearse os primeiros modelos relacionais, pero non é ata os anos 80 coa aparición dos primeiros PCs cando se populariza o emprego das bases de datos e se estandariza a linguaxe SQL. Nesta década hai un avance grazas ao reporting e á creación do concepto Datawarehouse, pero a gran maioría de proxectos acabou en fracaso debido a falta de comprensión á hora de deseñar e implementar un DWH, polo que comezaba a ser preciso o emprego de ferramentas específicas para a explotación das bases de datos. Durante esta década, aparecen as primeiras follas de cálculo que tratan moitos datos sen ningún tipo de vinculación entre sí.

Nos anos 90 estandarízase a correcta integración dos datos e aparecen a BBDD distribuídas grazas a arquitectura cliente/servidor. Comezan a aparecer ferramentas BI, pero que non son capaces de analizar grandes cantidades de datos e con numerosas limitacións e altos prezos.

Coa chegada do novo milenio, nace Hadoop e revolucionanse os prezos. Existe a capacidade de almacenamento masivo con alta velocidade de resposta e modularización de información estructurada e non estruturada.

Na actualidade existen numerosas formas de deseñar solucións BI, dende o BI tradicional ao Big Data ou Cloud BI, que nos permite unha solución totalmente integrada na nube.

1.3 Problema a resolver

Os proxectos dentro da organización na que se traballa están xestionados mediante Azure Devops, que inclúe ferramentas de desenvolvemento de software, colaboración, medición e xeración de informes, que se integran ampliamente nas aplicacións do lado servidor e do lado do cliente.

Debido ao incremento do número de proxectos a xestionar e o gran volume dos mesmos, considérase necesaria a implementación dun sistema de Business Intelligence para poder obter así información, e polo tanto coñecemento, acerca dos proxectos da organización.

1.4 Solución proposta

A solución proposta con este traballo é a realización dun Cadro de Mando interactivo dende o cal se poida obter información de maneira visual sobre todos os aspectos dos proxectos que se están a desenvolver. Para a realización do Cadro de Mando, empregáronse os datos que previamente son procesados mediante un modelo tabular.

Os datos obtéñense dende Azure Devops de dúas formas diferentes. Parte dos datos son recopilados directamente dende SSAS con oData, e o resto mediante API Rest cun servizo que

se executa periodicamente e xera nun servidor local diferentes arquivos CSV que posteriormente son cargados ao modelo tabular.

Unha vez importados os datos ao modelo, creamos os indicadores a medir mediante o uso de fórmulas DAX en SSAS, as relacións entre as táboas, e os diferentes roles.

Finalmente, en PowerBI deseñamos un cadro de mando onde se poden visualizar todos o datos acerca dos proxectos que se están a desenvolver.

1.5 Viabilidade da solución proposta

1.5.1 Viabilidade económica

De cara a determinar a viabilidade económica do proxecto proposta, compre analizar como mínimo os seguintes puntos:

- **Custo de persoal:** Este proxecto foi pensado para ser realizado por unha única persoa, cunha dedicación aproximada de 404 horas e cun 25% do traballo en horario laboral. Como dito traballo será empregado pola organización, en todo momento se considerou como viable.
- **Software:** O software empregado é gratuíto, polo que non foi un impedimento para a realización do proxecto. A única parte que supoñía custo eran as contas de PowerBI, pero xa se dispoñía delas ao empregarse para outros proxectos.
- **Hardware:** Non supuxo ningún custo a nivel de software, xa que o ordenador e servidor empregados para realizar este traballo xa estaban a disposición da organización para levar a cabo outros proxectos.

1.5.2 Viabilidade comercial

A realización deste proxecto serviu de base para a realización doutro proxecto, este sí a nivel comercial, onde se empregaron as mesmas ferramentas e unha arquitectura moi similar a esta, simplemente substituíndo o servizo por un pequeno ETL, xa que foi preciso un maior trato dos datos procedentes dunha BD.

1.5.3 Viabilidade conceptual

Como conclusión xeral, podemos dicir que é viable, xa que se pretende que a organización obteña beneficios á hora de realizar a xestión. Tamén serviu de base para a realización doutros proxectos.

No Anxeo B realízase unha estimación real sobre o que supoñería a realización deste proxecto cunha dedicación total dentro do horario laboral e con diferente persoal da organización.

1.6 Estrutura da memoria

A estrutura da memoria está formada por 7 capítulos que se detallan a continuación:

- **Capítulo 1:** Pequena introdución sobre o Business Intelligence, o problema a resolver, a solución que se propón e a viabilidade desta.
- **Capítulo 2:** Detállanse os compoñentes de Business Intelligence, as diferentes ferramentas dispoñibles e as plataformas con maior relevancia.
- **Capítulo 3:** Breve descrición das ferramentas e tecnoloxías empregadas.
- **Capítulo 4:** Especificación de requisitos e arquitectura do sistema.
- **Capítulo 5:** Implementación tanto do servizo de obtención de datos como do modelo tabular.
- **Capítulo 6:** Expónse o cadro de mando creado para este proxecto detallando os pasos seguidos para o seu desenvolvemento.
- **Capítulo 7:** Extráense algunhas conclusións do proxecto e defínense futuras liñas de traballo.

Capítulo 2

Estado da arte

Neste segundo capítulo explícanse os compoñentes de Business Intellegence, as ferramentas dipoñibles na actualidade e a relevancia das plataformas.

2.1 Introdución

Nos últimos anos, o aumento exponencial de tarefas a realizar nas consultorías de TI obrigou a migrar os proxectos a metodoloxías áxiles na nube para poder xestionalos de maneira eficiente. No caso da empresa na que se traballa, emprégase Azure Devops que ofrece integración, implementación e entrega contínua, mellora a seguridade e a conformidade e incrementa a confiabilidade e a repetitividade.

Antes que comezar coa migración dos proxectos a Azure Devops, estudouse a posibilidade de realizar unha análise máis profunda que a que nos ofrece Azure Log Analytics e Azure Monitor.

Tras dito análise, tomouse a decisión de crear sistema de Business Intelligence que consiste nun modelo tabular que será explotado de forma visual mediante a ferramenta de Power BI.

2.2 Compoñentes de Business Intelligence

2.2.1 Esquema xeral dunha solución de BI

Unha solución BI parte de diferentes fontes de orixe (BBDD, nube, ERPs...) e polo tanto, é preciso realizar unha fase de extracción, transformación e carga de datos (Extraction Transformation Load).

A fase de ETL adoita apoiarse nun almacén intermedio (xeralmente un Data warehouse), cuxo principal obxectivo consiste en evitar a saturación dos servidores funcionais da organización. Esta información almacénase no Data Warehouse corporativo e pode servir de base para a creación de Datamarts departamentais.

Un Datamart trátase dunha base de datos especializada, departamental e orientada a satisfacer as necesidades dun grupo de usuarios particular. Caracterízanse por conter a estrutura óptima para o análise dos datos de esa área da empresa, ben sexa mediante bases de datos transaccionais (OLTP) ou bases de datos analíticas (OLAP).

Tanto os datos dun Data Warehouse como dun DataMart, son explotados empregando ferramentas de análise e reporte. Nestas ferramentas baséase tamén a creación de produtos de BI máis complexos, como poden ser os sistemas de soporte á decisión (DSS), os cadros de mando (CMI ou BSC) e os sistemas de información executiva (EIS).

2.2.2 Data Warehouse

Trátase dunha tecnoloxía de almacenamento de datos desenvolta para optimizar o uso da información. Enfócase en optimizar a integración dos datos que proveñen de distintas fontes funcionais para logo procesala permitindo a súa análise dende infinidade de perspectivas e con grandes velocidades de resposta.

Nunha base de datos operacional almacénanse todas as transaccións da organización, tanto útiles como non útiles, mentras que nos DW só contemos información de interés que se require analizar.

O principal obxectivo do almacén de datos é extraer rendemento da información almacenada, é dicir, extraer os datos para unha análise posterior que axude a tomar decisións, mostrando un enfoque diferente respeto ás bases de datos convencionais.

2.2.3 Características Data Warehouse

Para levar a cabo un tratamento adecuado da información, o almacén de datos debe cumprir un conxunto de características:

- **Orientado ao tema:** Xa que non se coñecen os requerimentos dos usuarios no momento no que se constrúe o almacén de datos, a información non se estrutura segundo a súa funcionalidade (o uso que se lle vaia a dar), senón dividida por temas de interese.

- **Integración de datos:** Debe de garantizarse a calidade dos datos integrados e realizar unha xestión áxil de fontes con altos volumes de datos. Un compoñente que pode axudar á integración son os metadatos.

- **Información histórica e non volátil:** A historicidade indica cando se produce un acontecemento no mundo real. A súa importancia radica no momento de analizar a evolución no tempo. Calquera dato no almacén de datos debe ir acompañado do seu período de validez.

A non volatividade implica que non existan as operacións de modificar e borrar propiamente ditas. Os datos non se borran ou modifican, se non que se insertan correcións e a data na que se rexistrou.

2.2.4 Ventaxas e inconvenientes do DWH

Ventaxas:

- Facilita o acceso aos datos: Os usuarios poden acceder a unha gran cantidade de información, e polo tanto solventar un gran número de problemas que probablemente incrementen os beneficios para a organización.
- Costes mínimos: Reduce os custos relacionados coa informática nas empresas.
- Información consolidada: Un dos principais obxectivos é o de ser un repositorio central de información corporativa que pode ter como orixe diversos sistemas.
- Facilidade de funcionamento: Os almacéns de datos poden traballar en conxunto, e polo tanto, aumentar o valor operacional das aplicacións empresariais, en especial a xestión de relacións con clientes.
- Capacidade de aprendizaxe: Proporciona a capacidade de aprender dos datos do pasado e de predecir situacións futuras en diversos escenarios.

Inconvenientes:

- A implementación dun DW implica un alto custo e precisa dun mantemento.
- Non é moi útil para a toma de decisións en tempo real.
- Deseño complexo e multidisciplinar que require limpeza contínua, transformación e integración de datos.

2.2.5 Arquitectura xeral DW

Unha arquitectura Data Warehouse está formada polos seguintes compoñentes que interactúan entre sí:

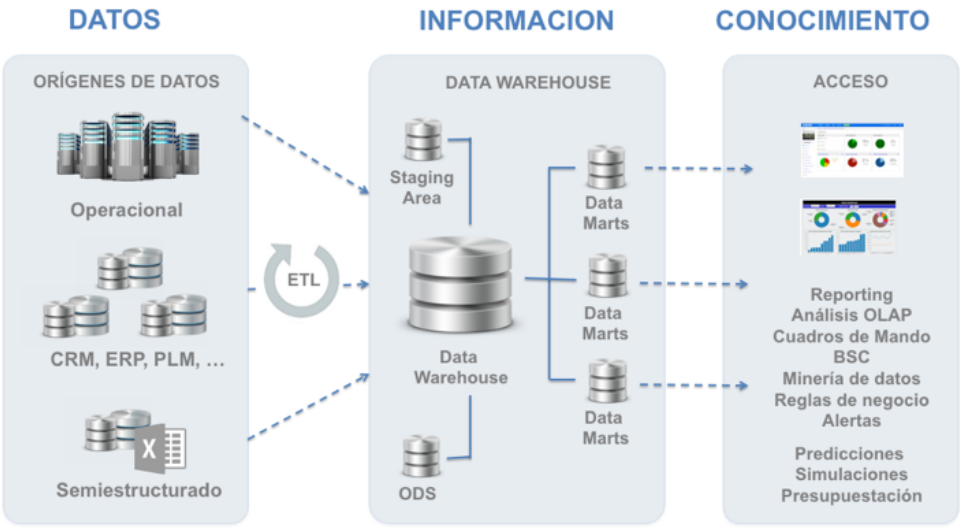

Figura 2.1: Arquitectura DataWareHouse [1].

- **Orixe de datos:** Os datos proceden de diversas fontes que as organizacións empregan diariamente.
- **Sistema ETL:** Realiza as funcións de extracción, transformación e carga dos datos.

  - **Extracción:** Xeralmente, é precisa a fusión de datos de diferentes orixes e con estruturas diferentes. A tarefa de extracción, converte todos estos datos a un formato adecuado para o inicio da transformación.

  - **Transformación:** Aplícanse unha serie de funcións sobre os datos co obxectivo de convertilos en datos preparados para a carga (tradución e codificación de códigos, obtención de valores calculados, xeración de novos campos, división da información e unión de datos de múltiples fontes).

  - **Carga:** Os datos transformados cárganse no almacén de datos. Aplicaranse todas as restricións e disparadores creados na base de datos de destino.
- **Almacén de datos:** Contén datos históricos e está orientado á explotación analítica dos datos.
- **ODS (Operational Data Store):** Dá soporte aos sistemas operacionais. Os datos recóllense dende a Staging Area e a súa actualización non é instantánea.

- **Staging Área:** Área temporal onde se recopilan os datos que se precisan dos sistemas de orixe e aplícanse as mínimas transformacións aos mesmos.
- **Data Marts:** Destinado a satisfacer as necesidades dun segmento de negocio en particular logrando un axuste máximo ao propósito dos usuarios da unidade de negocio.

2.2.6 Deseño e construción dos Data Warehouse/Datamarts

Os Data Warehouse trátanse dunha base de datos corporativa que replica os datos transacionais despois das fases de depuración e estruturación para actividades de consulta e reporte. Deste xeito, a información non se obtén directamente dende sistemas operacionais.

O deseño e estruturación incorrecta dos Data Warehouse ou Data Marts ocasionará problemas que poden levar a fracaso calquer esforzo posterior.

- Requirimentos: Especificación clara e precisa sobre as funcións que se esperan obter do Data Warehouse. Deben de analizarse dende varias perspectivas.
- Análise: Trata a conversión dos requerimentos en especificacións que sirvan de base para o deseño. Defínense os modelos lóxicos dos datos, os procedementos de conexión e as ferramentas de acceso ao usuario final.
- Deseño conceptual: Modelado do sistema facendo uso de modelos tales como o Entidade/Relación.
- Deseño lóxico: Realización do modelado multidimensional da base de datos.
- Deseño físico: Defínese o esquema a seguir, as ferramentas OLAP e deséñase o ETL.

O deseño conceptual pasa por diferentes fases durante o desenvolvemento dunha plataforma BI:

- Fase de construción do almacén de datos: Toman relevancia aspectos de estruturación da información
- Fase de implantación de ferramentas de soporte a alta dirección: Realízase a análise de criterios directivos (KPIs, factores de seguemento).

As técnicas de deseño máis importantes son os esquemas en estrela e copo de neve:

- **Estrela:** É o modelo máis empregado e o que presenta a estrutura máis sinxela. A táboa de feitos sitúase na parte central e está relacionada con diferentes dimensións.

Neste modelo, a única táboa que ten unha relación con outra é a de feitos, o que significa que toda a información relacionada cunha dimensión debe estar nunha soa táboa

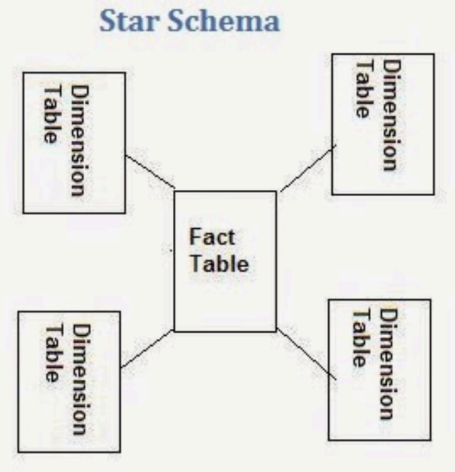

Figura 2.2: Modelo en estrela [2].

- **Copo de neve:** Trátase dunha variación ou derivación dun modelo en estrela. Neste modelo existen outras táboas que se relacionan coas dimensións e que non teñen relación directa coa táboa de feitos. Este tipo de modelo é máis difícil de manter que o modelo en estrela.

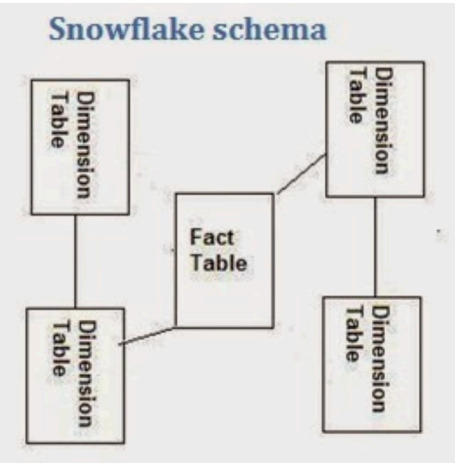

Figura 2.3: Modelo en copo de neve [2].

Ambos modelos empregan dous tipos de táboas que se explican a continuación:

- **Táboa de Feitos:** Táboa onde se almacenan todas aquelas medidas numéricas que incluiremos no noso sistema de Businesss Intelligence e que contén as chaves subrogadas

daquelas dimensións que definen o seu nivel de detalle e os indicadores. Ditas chaves no seu conxunto forma a chave primaria da táboa de feitos. Son táboas moi grandes con posibles datos redundantes e adoitan estar desnormalizadas.

- **Táboa de Dimensións:** Almacena información descritiva sobre os valores numéricos dunha táboa de feitos.

  Cada táboa de dimensións conten varias columnas e atributos que se utilizan para describir os procesos de negocio, pero só un deles forma parte da súa chave primaria.

  Para a creación das dimensións debemos de definir a granularidade, identificar as dimensións, crear relacións un a moitos e empregar dimensións compartidas.

2.2.7 Análise OLAP

Trátase dunha xeración de análise multidimensional con carácter dedutivo e coa posibilidadet de incorporar filtros personalizados a nivel de usuario.

O análise multidimensional consiste en combinar distintas áreas da organización, para obter información acerca do comportamento do negocio.

Os puntos de cada unha das dimensións poden agruparse por niveis seguindo unha certa xerarquía, de feito que un conxunto de puntos dun certo nivel forman outro punto no seu nivel superior.

As ferramentas OLAP ofrecen as seguintes opcións de navegabilidade:

- **Segmentar:** Facer unha segmentación dos datos dos que dispoñemos.

- **Profundizar:** Emprégase a ferramenta Drill Down, e con ela podemos bucear pola composición dos datos.

- **Sintetizar:** Faise uso da ferramenta Drill Up. Neste caso realízase o contrario que na anterior, trátase de ir accedendo dende niveis máis sinxelos a máis complexos.

- **Rotar:** Emprégase a ferramenta Drill Anywhere, para poder pasar dunha característica a outra pertenecente a unha xerarquía diferente.

- **Filtrar:** Tal e como o seu nome indica, o seu uso permite a creación de informes aplicando distintos filtrados de datos.

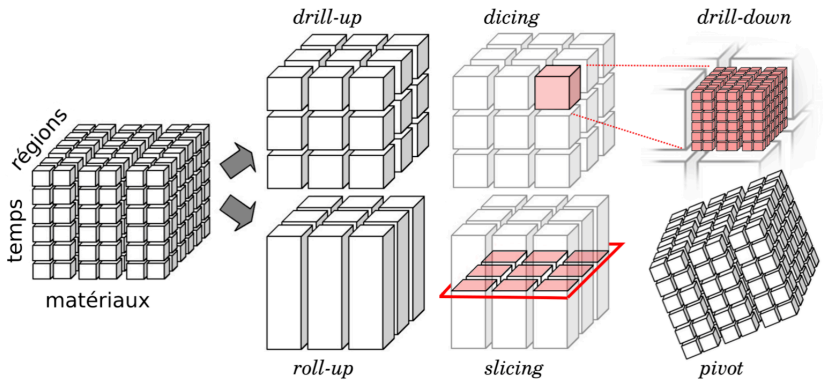

Figura 2.4: Navegabilidade OLAP [3].

2.3 Ferramentas de Business Intelligence

A continuación amósanse as ferramentas básicas de explotación da información:

- **Minería de datos (Data Mining):** Trátase de ferramentas de extracción de coñecemento útil.

  O obxectivo é perseguir patróns ocultos, tendencias e correlacións, e presentar esta información de maneira sinxela e accesible aos usuarios finais. Incorpora o emprego de tecnoloxías baseadas en redes neuronais, árbores de decisión, regras de indución, análise de series temporais e visualización de datos.

- **Cadro de mando analítico (EIS tradicionais):** Solucións que permiten visualizar, de forma rápida e sinxela, o estado dunha determinada situación empresarial, presente ou pasada, e que permite detectar anomalías e oportunidades.

- **Sistemas de Soporte de decisión (DSS):** Básease en modelos matemáticos e permiten axudar á toma de decisións. É unha ferramenta de Business Intelligence enfocada á análise dos datos dunha organización.

- **Query e Reporting:** Compónse de ferramentas para a realización de informes e listados, tanto en detalle como sobre información agregada, a partir da información dos almacéns de datos.

- **Cadro de Mando Integral (CMI) ou Balanced Scrorecard (BSC):** Utilizada no control empresarial. Permite establecer e monitorizar os obxectivos dunha empresa e das súas diferentes áreas ou unidades.

- **OLAP:** Ferramentas que manexan cuestións complexas de bases de datos relacionais, proporcionando un acceso multidimensional aos datos, capacidades intensivas de cálculo e técnicas de indexacións especializadas.

  Permiten aos usuarios trocear os datos, planteando consultas sobre diferentes atributos ou eixes. Empregan un servidor intermedio para almacenar os datos multidimensionais precalculados, de forma que a explotación sexa rápida.

- **Sistemas para a xestión do coñecemento (KMS):** Tecnoloxía que facilita o acceso á información corporativa.

2.4 Selección de ferramentas de Business Intelligence

Para a selección das ferramentas a empregar, deben de terse en conta diferentes aspectos:

- Información que se precisa e para qué: Non deben crearse indicadores e modelos complexos.
- A quen vai dirixida: Departamento de xestión, alta dirección, etc.
- Aspectos técnicos: Tempos de resposta, integración, seguridade, etc.
- Aspectos funcionais: Entorno gráfico, navegación, etc.

2.5 Relevancia das plataformas de Business Intelligence

Para a determinar as plataformas máis relevantes, emprégase o famoso Cadrante de Gartner. Este é elaborado por unha empresa de consultoría e investigación do mercado das novas tecnoloxías, adicada a investigar e analizar as tendenzas do mercado. A partir das conclusións obtidas, crea un ranking dos fabricantes con mellores solucións e produtos.

O Cadrante de Gartner divídese en:

- **Líderes:** Aquí atópanse os proveedores con maior puntuación. Estas empresas ofertan unha solución de produtos amplia e completa, e con capacidade de evolución según a demanda do mercado. Como se mostra na imaxe, neste cadrante están Microsoft, Tableau e Qlick.
- **Retadores ou aspirantes:** Proveedores con boas funcionalidades pero menor variedade de produtos (Microstrategy).
- **Visionarios:** Teñen capacidade para anticiparse ás necesidades do mercado pero non dispoñen de medios suficientes para realizar implantacións globais (Sisense, SSAS, SAP, etc).

- **Xogadores de nicho:** Atópanse en último lugar xa que non chegan a puntuar o suficiente en ningunha das dúas categorías (Oracle, Domo, Birst, etc).

A continuación móstrase como están posicionadas a nivel mundial as solucións de Businesss Intelligence en base a unha serie de conceptos como usabilidade e visión entre outros.

Figura 2.5: Cadrante máxico de Gartner [4].

Podemos observar na imaxe, como as ferramentas de Microsoft para Business Intelligence se sitúan na parte alta do cadrante de líderes.

Capítulo 3

Ferramentas e tecnoloxías empregadas

No terceiro capítulo explícanse de maneira detallada cada unha das ferramentas e tecnoloxías empregadas para a realización deste proxecto.

3.1 SQL Server Management Studio

Entorno integrado para a administración de infraestruturas de SQL. Proporciona ferramentas para configurar, supervisar e administrar instancias de SQL Server e bases de datos.

Neste caso empregaremos SSMS, instalado no servidor da organización para administrar a instancia tabular.

3.2 Visual Studio

Entorno de desenvolvemento integrado para Windows, Linux e MacOS. Compatible con múltiples linguaxes de programación e entornos de desenvolvemento web. Permite aos desenvolvedores crear sitios e aplicacións web, servizos web, aplicacións que se comuniquen entre estacións de traballo, páxinas web, dispositivos móviles...

Para a creación desde proxecto, engádese unha extensión de Visual Studio adicada a Business Intelligence (SQL Server Data Tools). Deste xeito, dentro do mismo IDE, tense a capacidade de carga de información no DWH e a de reporting, imprescindibles á hora de realizar un proxecto de BI.

SQL Server Data Tools (SSTD) inclúe ferramentas de creación de proxectos de intelixencia empresarial e modelos de proxectos para SQL Server Análisys Services, Reporting Services e Integration Services.

3.3 Devops

Combinación de persoas, procesos e tecnoloxías para permitir a entrega contínua de valor aos clientes.

Trátase dunha práctica de desenvolvemento de software que unifica o desenvolvemento e as operacións de TI. Significa coordinación e colaboración entre disciplinas que antes estaban illadas. Equipos de seguridade e de enxeñaría da calidade tamén forman parte do equipo máis amplo no modelo DevOps.

DevOps inclúe prácticas principais, como plantexamento e seguemento, desenvolvemento, compilación e probas, entrega, supervisión e operacións. Estas prácticas, xunto coas ferramentas e tecnoloxías de DevOps, permiten automatizar o ciclo de vida das aplicacións. Os procesos que soen ser manuais e lentos para os equipos, como actualizar o código ou aprovisionar un novo entorno, pódense facer de forma rápida e continua cando se utilizan ferramentas e prácticas de DevOps. Además, é máis sinxelo cumprir as normas de seguridade e confiabilidade, porque estas consideracións están integradas no proceso.

Con DevOps, disponse de todo o preciso para a entrega de mellores produtos en menos tempo. Ao reunir a persoas, procesos e tecnoloxías con prácticas y ferramentas compartidas, obtéñense as ventaxas de tempos de desenvolvemento reducidos, menor tempo de comercialización e maior calidade dos produtos.

3.3.1 Azure Devops

Trátase do sucesor de Visual Studio Team Services. Os seus servizos inclúen Azure Pipelines, Azure Boards, Azure Artifacts, Azure Repos, Azure Test Plans [7].

En Azure Devops, os proxectos poden ser públicos (como por exemplo GitHub, onde calquera pode consultar e colaborar) ou privados (para permitir acceder só a usuarios asignados por parte do administrador).

Para a creación dos proxectos, debemos de empregar os modelos dos que dispón Azure Devops, que son Basic, Agile, Scrum, e CMMI. Na nosa organización os proxectos a analizar empregan a metodoloxía Scrum, onde existe un só proxecto chamado Portfolio, que consta de diversos subproxectos creados como Backlogs. Cada un destes subproxectos, consta de múltiples tarefas, test case, bugs (que á sua vez conteñen outras tarefas)...

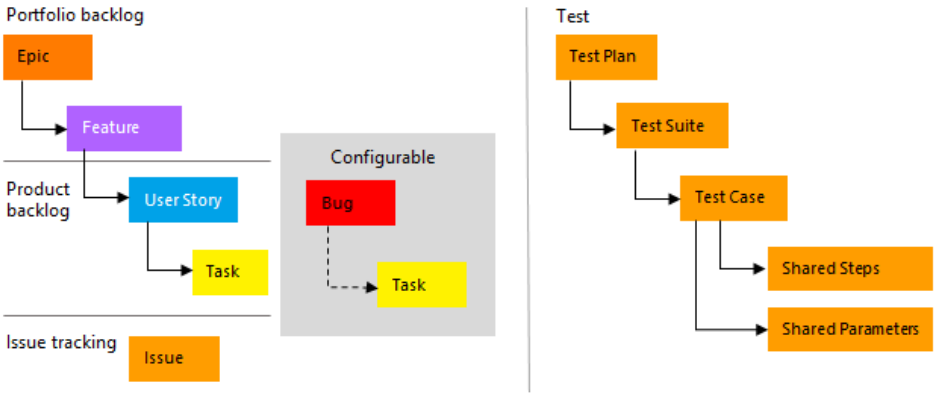

Figura 3.1: Compoñentes Azure Devops [5].

3.4 Analysis Services

Segundo a súa definición formal, Analysis Services é un motor de datos analítico empregado no soporte de decisións e análise de negocios. Proporciona modelos de datos semánticos de nivel empresarial para informes comerciais e aplicacións cliente.

3.5 Power BI

Trátase dunha solución destinada á intelixencia empresarial que permite unir diferentes fontes de datos, modelalos e analizalos para posteriormente, presentarlos a través de paneis e informes que poidan ser consultados dunha maneira sinxela e intuitiva.

Dispón de apps para todas as plataformas móviles para que os datos poidan ser consultados dende calquer lugar a través dun smartphone ou tablet.

3.6 Porta de enlace de datos local

Sistema que permite a conexión das orixes de datos locais ao servizo Power BI.

Aplicación que se executa no equipo local e emprega unha programación predeterminada para conectarse aos datos, recopilar actualizacións e insertarlas no servizo Power BI.

Proporciona unha transferencia de datos rápida e segura entre os datos locais e varios servizos na nube de Microsoft (Power BI, PowerApps, Microsoft Flow, Azure Analysis Services e Azure Logic Apps) [8].

Ao empregar unha porta de enlace, as organizacións poden manter as orixes de datos nas súas redes locais e empregar de forma segura esos datos en servizos da nube.

Tipos de portas de enlace:

- **Porta de enlace de datos local:** Permite que varios usuarios se conecten a varias orixes de datos locais.
- **Porta de enlace de datos local (modo persoal):** Permite a un usuario conectarse ás orixes e que non poida compartilos con outros usuarios. Este tipo, só se pode empregar con Power BI.

3.7 oData

Protocolo equivalente web do estándar SQL para bases de datos. De igual xeito que SQL é unha linguaxe que permite solicitar datos filtrados, ordenados e proxectados dende un DBMS, oData é unha sintaxe que permite as mesmas operacións, pero sobre fontes de datos que están a disposición a través do protocolo HTTP.

O cliente envía unha petición seguindo a sintaxe oData e recibe o resultado. Pode tratarse de calquer cliente web, como por exemplo un navegador.

3.8 API Rest

Estándar lóxico e eficiente para a creación de servizos web.
Restricións dun sistema RESTFull:

- Cliente-servidor: Cliente e servidor debilmente acoplados.
- Sin estado: Non é preciso manter sesións.
- Cacheable: Debe admitir un sistema de almacenamento en caché.
- Interface uniforme: Interface xenérica para administrar as interaccións cliente-servidor de forma uniforme. Cada recurso do servizo REST debe ter unha única dirección, "URI".
- Sistema de capas: O servidor pode dispor de varias capas para a súa implementación, polo que se mellora a escalabilidade, o rendemento e a seguridade.

3.9 DAX (Data Analysis Expressions)

Recopilación de funcións, operadores e constantes que poden ser empregados nunha fórmula ou expresión para calcular e devolver un ou varios valores. Trátase dunha linguaxe de expresións, creada para traballar con Modelos tabulares (Power Pivot, SQL Server Analysis Services Tabular, Power BI).

Existen 8 tipos de datos DAX: Número Enteiro, Número Real, Boolean, String, Date, Moeda, BLANK e Table.

A linguaxe DAX, presenta unha sintaxe sinxela:

- As fórmulas comenzan co signo '='.
- As expresións conteñen funcións, operadores, constantes e referencias a columnas.
- Os nomes das columnas escríbense entre corchetes.

Capítulo 4

Requisitos e arquitectura do sistema

DURANTE o desenvolvemento deste capítulo detállanse os distintos tipos de requisitos e a arquitectura do sistema.

4.1 Especificación de requisitos

Dirixida tanto ao cliente como ao equipo de desenvolvemento. A linguaxe empregada debe ser informal, para que sexa fácilmente comprensible para todas as partes involucradas no desenvolvemento, xa que esta parte está dirixida tanto ao cliente como ao equipo encargado de desenvolver o sistema.

4.1.1 Requisitos de usuario

Este tipo de requisitos normalmente son recollidos mediante diagramas que explican o que se pretende conseguir coa realización do software. Adoita empregarse a linguaxe natural para a comunicación entre o cliente e o equipo de desenvolvemento.

Para a realización deste traballo realizouse un pequeno esquema sobre a arquitectura xeral e unha definición dos indicadores que se desexan mostrar no cadro de mando.

4.1.2 Requisitos de sistema

Descrición máis detallada dos servizos exactos que se proporcionarán e as súas restricións. Estos requisitos serven como contrato co cliente. Á súa vez os requisitos de sistema poden dividirse en requisitos funcionais, non funcionais ou de dominio.

- **Requisitos funcionais:** Especificación sobre o que o sistema a desenvolver debe facer e os servizos que debe proporcionar.

  Neste caso os requisitos funcionais estaban bastante claros, xa que só se pretende que os usuarios accedan aos cadros de mando dende a web e poidan interactuar con eles

aplicando diversos filtros.

- **Requisitos non funcionais:** Propiedades do sistema que non están focalizadas no que o sistema debe facer, senón que están máis centradas en temas como a seguridade, a usabilidade, a consistencia e a fiabilidade.

4.2 Arquitectura do sistema

4.2.1 Arquitectura xeral

Durante o preríodo de desenvolvemento deste proxecto, seguiuse unha arquitectura similar a da seguinte imaxe:

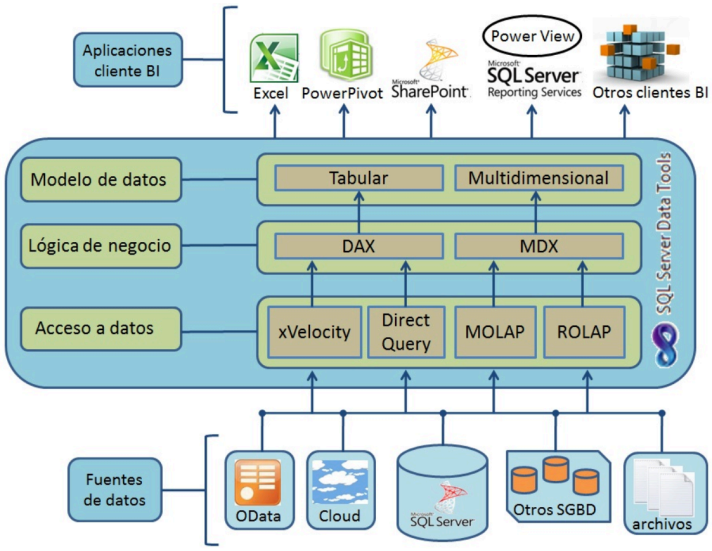

Figura 4.1: Arquitectura xeral do sistema [6].

- Parte superior: Aplicacións cliente, dentro das cales empregamos PowerBI.
- Parte inferior: Orixes de datos, que para este proxecto son oData e ficheiros de texto.
- Parte central: Modelo de datos, que para este traballo nos decantamos polo tabular, tal e como se explica no seguinte apartado.

4.2.2 Selección do modelo de datos

Para a realización dun modelo de Analysis Services, existen dúas plataformas:

- **Azure Analysis Services:** Modelos tabulares, DirectQuery, particións, seguridade a nivel de fila, relacións bidireccionais e traducións.

- **Sql Analysis Services:**
  - **Modelo Multidimensional:** Construcións de modelado OLAP (cubos, dimensións, medidas).
  - **Modelo tabular:** Construcións de modelado relacional (tablas, columnas...). Internamente, os metadatos hérdanse de construcións de modelado OLAP.
  - **Power Pivot:** Ofrece modelado de datos visuais en Excel, con soporte de servidor proporcionado a través de SharePoint. Presenta unha infraestrutura interna tabular.

No momento de definir o noso modelo SSAS, é preciso analizar profundamente os diversos usos e implementacións do sistema a desenvolver, xa que ambos presentan diferenzas importantes ao estar encamiñados a diferentes propósitos.

Debemos de ter en conta que o motor de almacenamento dos modelos tabulares realízase en memoria e persiste en disco, polo que garantiza a súa recuperación tras un reinicio.

Os modelos tabulares presentan unha arquitectura sinxela, polo que poden ser xerados de maneira máis rápida que os multidimensionais.

Non existe o concepto de agregacións tal e como ocorre nos multidimensionais, polo que ao realizarse distintas consultas, só traballará coas columnas que figuren nesta.

Á hora de decantarse polo modelo tabular, deben de terse presentes algunhas limitacións importantes:

- Non se permiten relacións moitos a moitos.
- Non é posible crear máis dunha relación entre duas táboas.
- Non está permitido o uso de relacións circulares.
- Á hora de realizar a extracción de datos, é preciso realizalo mediante unha única consulta por cada táboa do modelo

A seguinte táboa reflexa de maneira resumida as principais diferenzas entre ambos modelos:

	Multidimensional	**Tabular**
Tamaño	Grande	Mediano - Pequeno
Recursos HW	CPU, Memoria, Disco IO	CPU, Memoria
Tempo real	SI	SI
Linguaxes	MDX	DAX, MDX
Rapidez consulta	Rápido	Máis rápido
Complexidade desenvolvemento	Media - Alta	Media - Baixa
Complexidade deseño	Complexo	Máis sinxelo
Minería de datos	SI	NON
Licenza	Todas as versións	Enterprise/BI

Táboa 4.1: Diferenzas entre multidimensional e tabular.

Finalmente decantámonos por empregar un modelo de datos tabular, xa que o volume de datos a tratar non é tan grande como para precisar un modelo multidimensional, búscase velocidade de consulta e non unha excesiva complexidade á hora de modelar o sistema.

Capítulo 5

Implementación do sistema

Neste capítulo explícase como se desenvolveu o sistema. Consta de dúas partes: a implementación do servizo e a implementación do modelo tabular.

5.1 Implementación do servizo

5.1.1 Creación do token de acceso

Para a obtención de datos, creouse en AzureDevops un token de acceso. Para a creación deste debemos acceder dende o navegador á configuración do perfil de usuario e posteriormente á seguridade.

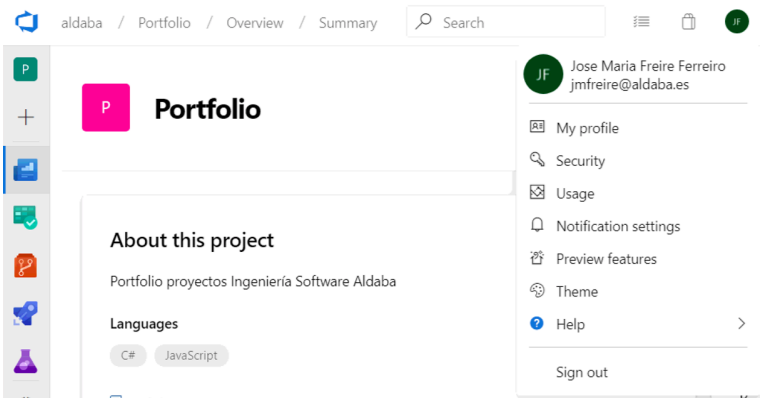

Figura 5.1: Paso 1. Token.

Unha vez dentro, podemos crear o token, asignándolle un nome, unha data de expiración (por exemplo 90 días) e uns permisos. Para este proxecto marcamos ´Full Access', xa que se pretende ter total acceso para a importación dos datos.

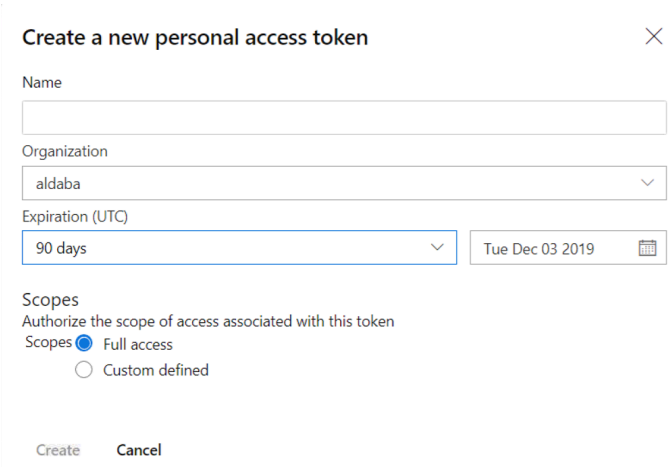

Figura 5.2: Paso 2. Token.

5.1.2 Creación do servizo API Rest

Debido a que mediante o protocolo Odata, non é posible obter parte dos datos a analizar, tomouse a decsión de crear un servizo API Rest en C# que obtén os datos periodicamente e os almacena en diferentes ficheiros planos no servidor local. Posteriormente serán importados ao modelo de SSAS.

Para a construción do servizo empréganse diferentes clases, xa que é preciso deserializar os JSON unha vez obtidos os datos mediante API REST [9]. A continuación mostramos un exemplo de obtención dos datos dunha das táboas (TestPlans). De forma moi similar, obtivéronse tamén os datos dos TestRun, Releases, Repositorios e Builds.

```
private static void OnTimedEvent(Object source, ElapsedEventArgs e){
    log4net.Config.XmlConfigurator.Configure();
    var url_plan = "https://dev.azure.com/aldaba/portfolio/_apis/testplan/plans?api-version=5.1-preview.1";
    try{
        HttpWebRequest request_Plan = (HttpWebRequest)HttpWebRequest.Create(url_plan);
        var credentials = Convert.ToBase64String(Encoding.ASCII.GetBytes(
                string.Format("{0}:{1}", "", token)));
        request.Headers.Add(HttpRequestHeader.Authorization, "Basic " + credentials);
        request.ContentType = "application/json";
        request.UserAgent = "Nothing";
        request_Plan.Headers.Add(HttpRequestHeader.Authorization, "Basic " + credentials);
        request_Plan.ContentType = "application/json";
        request_Plan.UserAgent = "Nothing";
        HttpWebResponse response_Plan = request_Plan.GetResponse() as HttpWebResponse;
            using (Stream responseStream = response_Plan.GetResponseStream()){
                StreamReader reader = new StreamReader(responseStream, Encoding.UTF8);
                string json_string = reader.ReadToEnd();
                string csv = jsonToCSV(json_string, ",");
                File.WriteAllText(@"\\aldvm061\shared\testPlans.csv", csv);
            }
    }
    catch (Exception ex){
        Logger.Error(ex);
        throw (ex);
    }
}
```

Figura 5.3: Obtención dos TestPlans.

O JSON a deserializar para este exemplo presenta a seguinte estrutura (moi similar ao resto de JSON pero con campos diferentes).

Este está composto por unha lista 'value' que contén todos os campos que se deseña recuperar.

Á súa vez, hai campos (como é o caso de project) que están compostos de outros campos, polo que será preciso crear outra clase.


```
{
"value":[
  {
    "id":1577,
    "project":{
       "id":"875leb9b-3293-40af-90e9-fa7d412992e1",
       "name":"Portfolio",
       "state":"unchanged",
       "visibility":"unchanged",
       "lastUpdateTime":"0001-01-01T00:00:00"
    },
    "rootSuite":{
       "id":1578,
       "name":"DEV_07_Targaryen_Stories_18.33"
    },
    "_links":{
       "_self":{
          "href":"https://dev.azure.com/aldaba/Portfolio/_apis/testplan/Plans/1577"
       },
       "clientUrl":{
          "href":"mtms://dev.azure.com:443/aldaba/p:Portfolio/Testing/testplan/connect?id=1577"
       },
       "rootSuite":{
          "href":"https://dev.azure.com/aldaba/Portfolio/_apis/testplan/Plans/1577/Suites/1578"
       }
    },
    "revision":0,
    "name":"DEV_07_Targaryen_Stories_18.33",
    "areaPath":"Portfolio\\DEV_07_Targaryen\\Finsa",
    "iteration":"Portfolio\\FINSA\\FINSA-1803-POOL-IS\\18.33",
    "owner":null,
    "state":"Active"
  }
]
}
```


Figura 5.4: Estrutura JSON.

Para a súa deserialización, do JSON empregamos a seguinte clase, donde se deserializan os elementos da lista 'value'.

```
namespace JsonDownloader
{
    class TestPlans
    {
        public List<TestPlansComposition> value { get; set; }

    }
}
```

Figura 5.5: Paso 1. Deserialización JSON.

Para deserialización dos campos internos da lista, creamos a seguinte clase:

```
namespace JsonDownloader
{
    class TestPlansComposition
    {
        public int id { get; set; }
        public IdNameClass project { get; set; }
        public IdNameClass rootSuite { get; set; }
        public int revision { get; set; }
        public string name { get; set; }
        public string areaPath { get; set; }
        public string iteration { get; set; }
        public string state { get; set; }
    }
}
```

Figura 5.6: Paso 2. Deserialización JSON.

Como se explicaba no JSON e se mostra na clase anterior, existen campos que están formados á sua vez por outros. Neste caso trátase dos campos project e rootSuit, polo que creamos a maiores outra clase que tamén se empregará para outros campos do resto dos JSON:

```
6 referencias
class IdNameClass
{
    5 referencias
    public string id { get; set; }
    4 referencias
    public string name { get; set; }
}
```

Figura 5.7: Paso 3. Deserialización JSON.

Para a conversión do JSON a táboa empregamos o seguinte método:

```
public static DataTable jsonStringToTable(string jsonContent){
    DataTable table2 = new DataTable();
    table2.Columns.Add("id", typeof(string));table2.Columns.Add("Project_Id", typeof(string));
    table2.Columns.Add("Project_Name", typeof(string));
    table2.Columns.Add("Root_Suit_Id",typeof(string));
    table2.Columns.Add("Root_Suit_Name", typeof(string));
    table2.Columns.Add("Revision", typeof(string));table2.Columns.Add("name", typeof(string));
    table2.Columns.Add("AreaPath", typeof(string));
    table2.Columns.Add("Iteration", typeof(string));table2.Columns.Add("State", typeof(string));

    TestPlans datalist3 = JsonConvert.DeserializeObject<TestPlans>(jsonContent);

    foreach (var elemento in datalist3.value){
        table2.Rows.Add(new object[] {
            elemento.id.ToString(),elemento.project.id.ToString(),elemento.project.name.ToString()
            ,elemento.rootSuite.id.ToString(),elemento.rootSuite.name.ToString()
            ,elemento.revision.ToString(),elemento.name.ToString(),elemento.areaPath.ToString()
            ,elemento.iteration.ToString(),elemento.state.ToString()});
    }
    return table2;
}
```

Figura 5.8: Conversión a táboa.

Unha vez obtidos os datos das táboas indicadas anteriormente, podemos obter os das seguintes:

- TestsResults: Para a obtención destes datos é preciso dispoñer dos IDs dos TestsRuns e ir compoñendo a url.
- Commits e Pushes: De forma similar aos TestResults, pero nestes caso as url están compostas polos IDs dos repositorios.
- BuildsWorkitems: Como nos casos anteriores, só que a url está composta polos IDs das Builds.

A continuación móstrase como exemplo a obtención dos Commits. Foi preciso crear tamén as clases (RepositoryCommits, RepositoryCommitsComposition) para deserializar o JSON.

```
private static DataTable jsonStringToTable(string jsonContent)
{
    DataTable table2 = new DataTable();
    table2.Columns.Add("RepositorytId", typeof(string)); table2.Columns.Add("CommitId", typeof(string));
    table2.Columns.Add("Author_name", typeof(string));table2.Columns.Add("Author_mail", typeof(string));
    table2.Columns.Add("Comment", typeof(string));table2.Columns.Add("ChangeContsAdd", typeof(string));
    table2.Columns.Add("ChangeCountsEdit", typeof(string));table2.Columns.Add("ChangeCountsDelete", typeof(string));

    Repositories datalist2 = JsonConvert.DeserializeObject<Repositories>(jsonContent);
    List<string> idList = (from element in datalist2.value
                           select element.id).ToList<string>();
    foreach (string indent in idList)
    {
        var url = "https://dev.azure.com/aldaba/portfolio/_apis/git/repositories/" + indent + "/commits?api-version=5.1";
        string token = "rodgfsqbia6jxydgzkcfofw24wsgik7dejcdz5r5sciixgxsqplq";
        try
        {
            HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(url);
            var credentials = Convert.ToBase64String(Encoding.ASCII.GetBytes(
                    string.Format("{0}:{1}", "", token)));
            request.Headers.Add(HttpRequestHeader.Authorization, "Basic " + credentials);
            request.ContentType = "application/json";
            request.UserAgent = "Nothing";
            HttpWebResponse response = request.GetResponse() as HttpWebResponse;

            using (Stream responseStream = response.GetResponseStream())
            {
                StreamReader reader = new StreamReader(responseStream, Encoding.UTF8);
                string json_string2 = reader.ReadToEnd();
                RepositoryCommits datalist3 = JsonConvert.DeserializeObject<RepositoryCommits>(json_string2);

                foreach (var elemento in datalist3.value)
                {
                    table2.Rows.Add(new object[] {
                            indent,
                            elemento.commitId.ToString(), elemento.author.name.ToString(),
                            elemento.author.email.ToString(), elemento.comment.ToString(),
                            elemento.changeCounts.Add.ToString(),elemento.changeCounts.Edit.ToString(),
                            elemento.changeCounts.Delete.ToString()
                    });
                }
            }
        }
        catch (Exception ex)
        {
            Logger.Error(ex);
            throw (ex);
        }
    }
    return table2;
}
```

Figura 5.9: Obtención dos Commits.

Unha vez convertidos todos os json a formato táboa, debemos de pasalos a CSV. Para iso empregamos o seguinte método:

```
public static string jsonToCSV(string jsonContent, string delimiter)
        {
            StringWriter csvString = new StringWriter();
            using (var csv = new CsvWriter(csvString))
            {

                csv.Configuration.SkipEmptyRecords = true;
                csv.Configuration.WillThrowOnMissingField = false;
                csv.Configuration.Delimiter = delimiter;

                using (var dt = jsonStringToTable(jsonContent))
                {
                    foreach (DataColumn column in dt.Columns)
                    {
                        csv.WriteField(column.ColumnName);
                    }
                    csv.NextRecord();

                    foreach (DataRow row in dt.Rows)
                    {
                        for (var i = 0; i < dt.Columns.Count; i++)
                        {
                            csv.WriteField(row[i]);
                        }
                        csv.NextRecord();
                    }
                }
            }
            return csvString.ToString();
        }
```

Figura 5.10: Conversion a CSV.

Posteriormente debemos de instalar o servizo no servidor tal e como se indica no Anexo A. Do mesmo xeito que calquer outro servizo de Windows, debemos de pulsar en iniciar para que este comece a descarga dos datos.

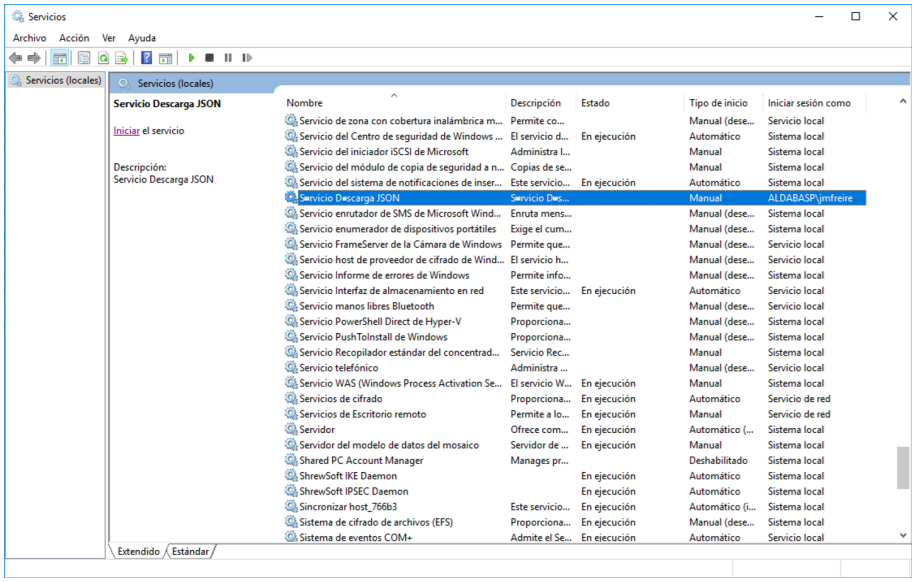

Figura 5.11: Iniciar servizo.

Finalmente programamos este servizo para que comece a executarse tras o reinicio diario do servidor.

5.2 Implementación do modelo Tabular

5.2.1 Creación do modelo Tabular

Para a implementación do modelo tabular emprégase a ferramenta Visual Studio (Business Intelligence). Debe de seleccionarse un dos seguintes modelos:

- **Proyecto tabular de Analysis Services:** Úsase para a creación dun modelo tabular dende cero.
- **Impotar do Servidor (Tabular):** Adoita empregarse cando se desexa crear un modelo tabular extraendo os metadatos dun existente en Analysis Services.
- **Importar dende Power Pivot:** Para a creación dun modelo tabular extraendo os metadatos e os datos dun archivo Power Pivot.

Para este caso concreto, seleccionamos a primeira das opcións xa que crearemos un modelo tabular dende cero.

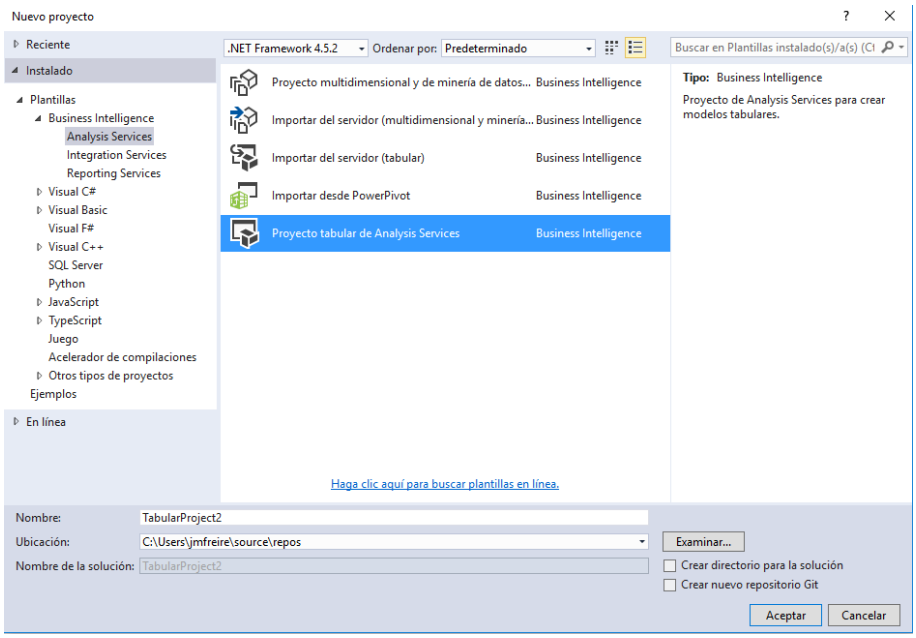

Figura 5.12: Creación proxecto Tabular.

Tal e como se mostra na imaxe inferior, debe de indicarse o servidor de Analysis Services que contén os metadatos do modelo a importar e o nivel de compatibilidade. Neste caso, empregamos o servidor facilitado pola empresa aldvm061.

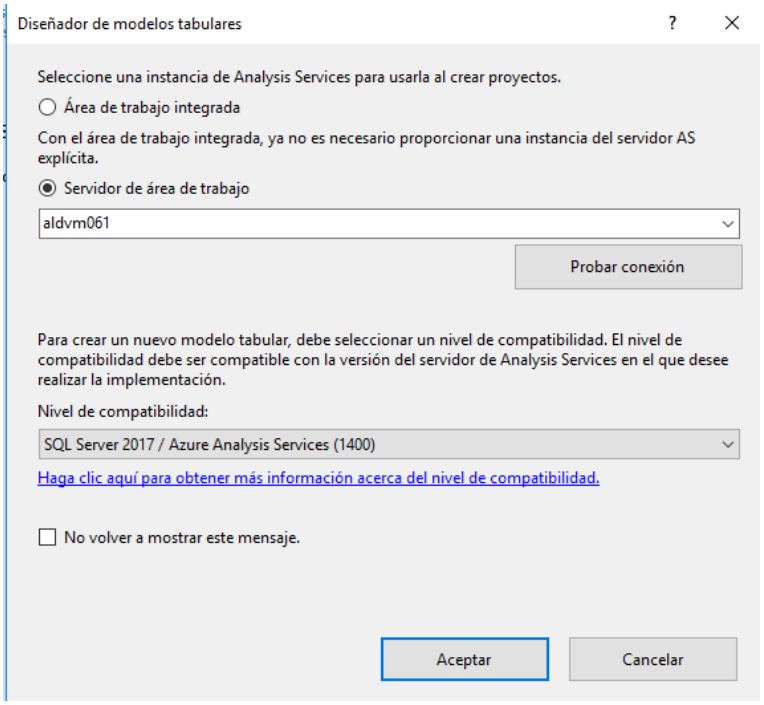

Figura 5.13: Asignación do servidor.

5.2.2 Importación de datos

A importación de datos en SSAS para este proxecto realízase dende tres orixes de datos distintas:

- AzureDevops.
- Ficheiros CSV.
- Táboa Excel.

Para a importación dende AzureDevops, debemos de seleccionar o compoñente oDta, no cal debe de indicarse a url correspondente e o token de acceso obtido previamente.

A outra orixe de datos son os ficheiros planos en formato .csv obtidos no servizo creado. Importaranse ao modelo do mesme xeito pero seleccionando 'Texto o CSV' en lugar de oData.

Por último debemos de importar unha táboa facilitada pola organización e contida nun ficheiro excel, onde se manexan os permisos dos usuarios das distintas áreas de traballo.

Mail	AreaId
ldiaz@aldaba.es	511162f5-eb4f-4d80-a685-3b1186465b11
ldiaz@aldaba.es	0f7637b6-9c76-46ee-acb8-cd980e3a6e23
ldiaz@aldaba.es	ef3c9b7d-5993-4dca-ae9e-aebc79c9f237
ldiaz@aldaba.es	a947fd5f-08c5-47cd-add9-aa2251c78818
jmartinez@aldaba.es	ef3c9b7d-5993-4dca-ae9e-aebc79c9f237
jtrigo@aldaba.es	ef3c9b7d-5993-4dca-ae9e-aebc79c9f237
malinares@aldba.es	9ac98290-bb89-4cf3-93e7-182f9424e68a
malinares@aldba.es	1a0c8bdd-ef73-4407-a84e-6d59ed3b663b
malinares@aldba.es	0f7637b6-9c76-46ee-acb8-cd980e3a6e23
malinares@aldba.es	fc2134fe-3732-4dfc-b36e-2fc19da5c7da

Figura 5.14: Táboa de permisos.

Para a importación das táboas indicadas dende Análysis Services debemos de acceder a *Modelo > Importar desde el origen de datos.* A continuación móstrase un exemplo de importación directa dende Azure Devops:

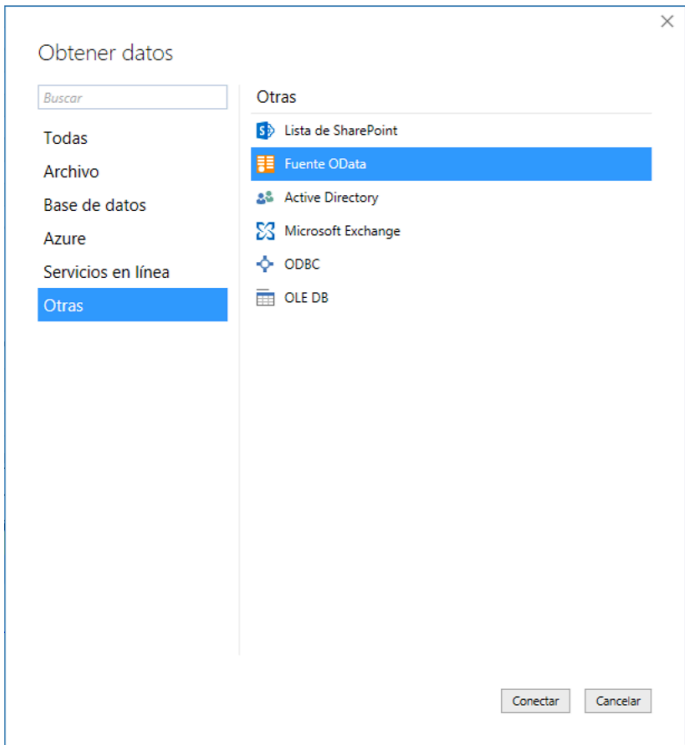

Figura 5.15: Importación de datos dende oData.

5.2.3 Relacións no modelo Tabular

Tras a importación de todas as táboas necesarias, debemos de modificar as relacións importadas directamente e incluír as novas ao modelo.

Para definir unha relación entre dúas táboas, unha delas ten que ter unha columna con valores que non se repitan e que sería a columna chave da táboa, e a outra debe ter unha columna do mesmo tipo e contendo os mesmos valores pero que poden estar repetidos (relación dun a moitos, xa que por cada fila dunha táboa poden existir moitas filas da táboa relacionada).

As relacións poden crearse tanto arrastrando dende unha táboa a outra as columnas a relacionar, como indicando as táboas e as columnas manualmente tal e como se mostra na imaxe:

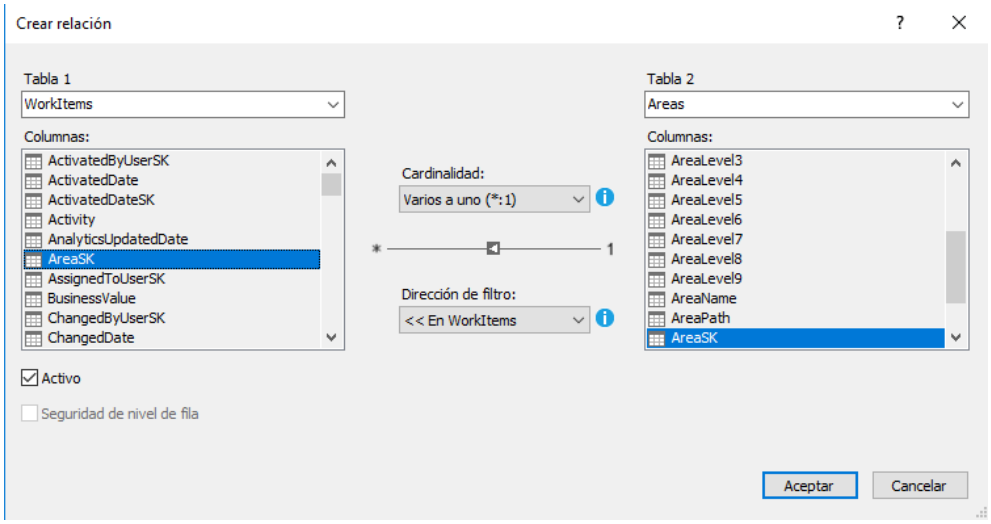

Figura 5.16: Relacións no modelo Tabular.

A esta ventá pode accederse seleccionando os menús *Tabla* > *Crear Relacións* ou facendo click dereito sobre unha columna dunha das táboas e seleccionando Crear Relación.

Unha vez creadas todas as relacións do modelo, as táboas quedarían relacionadas da seguinte maneira:

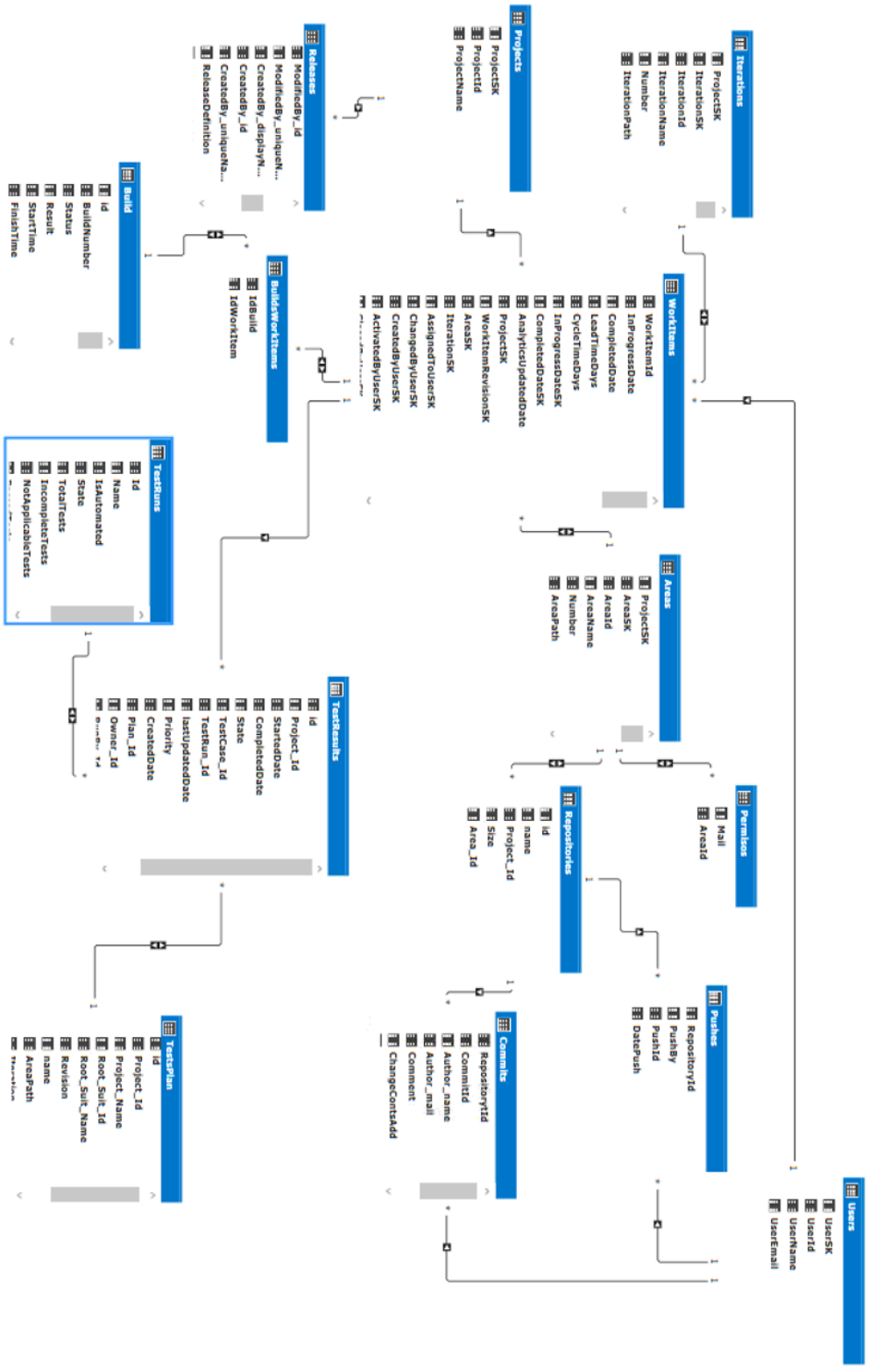

Figura 5.17: Modelo Tabular.

5.2.4 Ocultación de campos

Unha vez importadas todas as táboas ao modelo, pódese observar como moitos dos campos non son necesarios ou simplemente se empregan para realizar algún cálculo.

Estos campos que non desexamos propagar ao cadro de mando podemos ocultados tal e como se mostra na seguinte imaxe:

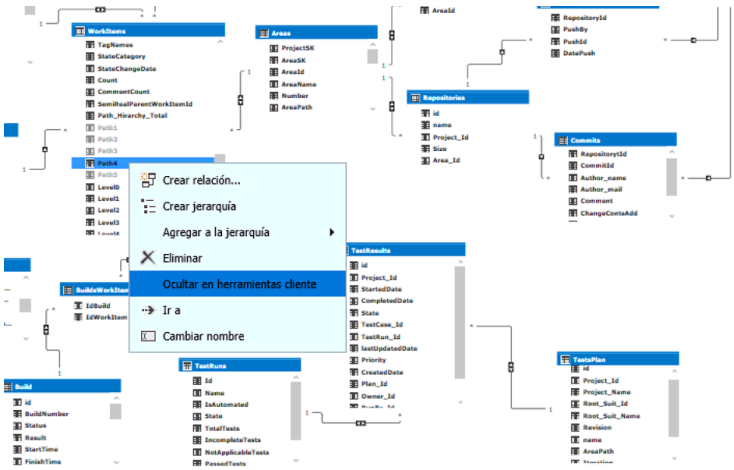

Figura 5.18: Ocultación de campos.

5.2.5 Creación de columnas calculadas

As columnas calculadas son un tipo especial de columna que conteñen unha expresión DAX que xeralmente usa datos doutras columnas para devolver un resultado. Este cálculo faise cando se importan os datos ao modelo e o resultado se almacena en cada fila deste.

A continuación móstrase como exemplo a construción da xerarquía (pais-fillos) entre os diferentes WorkItems. Orixinalmente, a táboa contiña para cada WorkitemId un pai almacenado na columna ParentWorkItemId.

Coa seguinte expresión, créase unha nova columna onde se inclúen só os pais que á súa vez son WorkItems. Este paso engádese para omitir os valores da columna ParentWorkItemId que non existen á súa vez como WorkItemID.

```
1  SemiRealParentWorkItemID = IF (WorkItems[ParentWorkItemId] in
    ALL(WorkItems[WorkItemId]);WorkItems[ParentWorkItemId];BLANK())
2
```

Unha vez creada a columna anterior, debe de obterse toda a xerarquía para cada WorkItem. Para iso, constrúese a seguinte columna, que obtén para cada fila un resultado similar a este '1129|341|351|372'.

```
Path_Hirarchy_Total =
PATH(WorkItems[WorkItemId];[SemiRealParentWorkItemId])
```

Posteriormente, pode extraerse cada elemento (Id) da lista obtida na expresión anterior. Deben crearse tantas columnas como número de elementos conteña a fila con máis niveis de xerarquía, polo que a seguinte expresión debe de repetirse ata o último dos niveis simplemente alternando a posición. A continuación móstrase un exemplo para obter o primeiro dos elementos:

```
Path1 = PATHITEM(WorkItems[Path_Hirarchy_Total];1;INTEGER)
```

Tras a obtención das columnas cos IDs dos diferentes niveis da xerarquía, coa seguinte expresión, pode obterse o título de cada un dos WorkItems. Como no caso anterior, debe de repetirse tantas veces como columnas creadas:

```
Level0 =
LOOKUPVALUE(WorkItems[Title];WorkItems[WorkItemId];[Path1])
```

5.2.6 Creación de medidas

As medidas son outro tipo de columna especial que tamén contén unha expresión DAX, pero neste caso, o cálculo só se realiza cando a columna é empregada nun informe e non se almacena no modelo. Xeralmente, as medidas agregan datos de varias filas da táboa.

Os nomes das medidas deben de ser únicos en todo o modelo, xa que ainda que só se definan para unha táboa, estas son globais.

Tanto para a creación das columnas calculadas como das medidas, empréganse expresións que poden conter operadores e chamadas a funcións.

A continuación inclúese algún exemplo de medida:

```
Build_Failed:= IF(
(CALCULATE(DISTINCTCOUNT(Build[id]);Build[Result]="failed")) =
BLANK();0;
(CALCULATE(DISTINCTCOUNT(Build[id]);Build[Result]="failed")))
```

```
Epic_en_curso:=CALCULATE(COUNT(
'WorkItems'[WorkItemId]);WorkItems[ClosedDate] = BLANK();
'WorkItems'[State] = "Active")
```

```
BugsPendientes:=CALCULATE(COUNT(
'WorkItems'[WorkItemId]);'WorkItems'[State] = "Active"
|| 'WorkItems'[State]="New";
'WorkItems'[WorkItemType] = "Bug")

```

5.2.7 Asignación de roles

A creación de Roles dende o modelo tabular pode realizarse tanto dende o SSMS (Sql Server Management Studio), como dende SSDT (Sql Server Data Tools).

A continuación mostramos cómo se asignan os roles aos usuarios dende SSTD e como se inclúen filtros a nivel de fila dependendo do rol asignado.

Para a creación de Roles, debemos de acceder ao menú *Modelo* > *Roles*

Nesta ventana engádese o novo rol e asígnanselle permisos, neste caso de lectura. Para este exemplo tamén se pode incluír unha sentenza DAX, onde se indica que os usuarios conectados terán acceso aos datos das áreas indicadas na táboa de permisos.

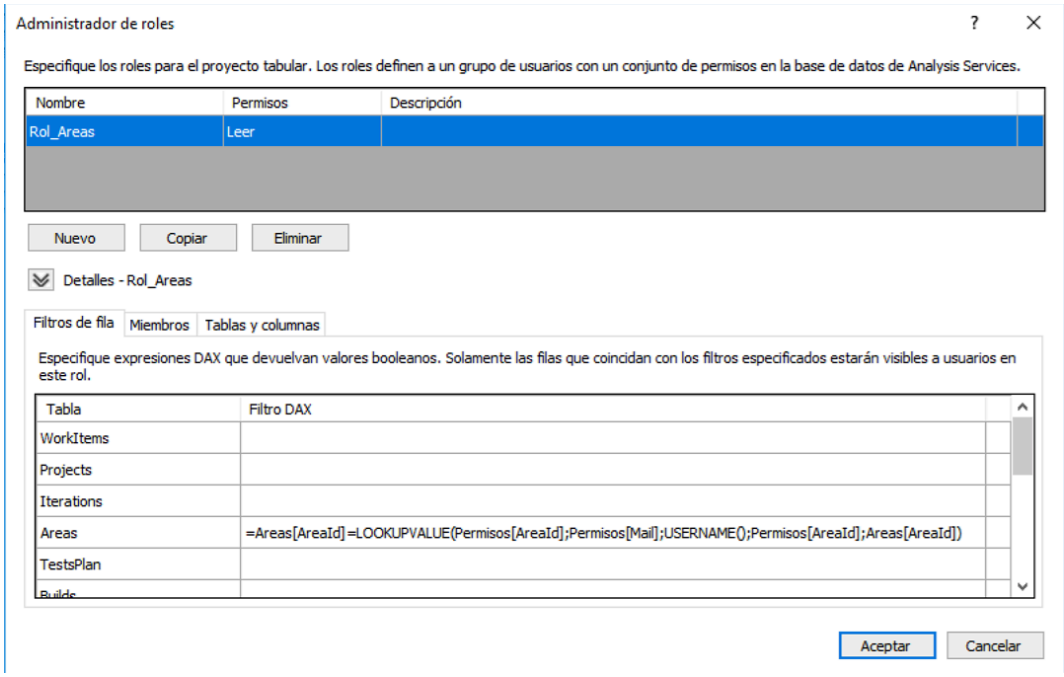

Figura 5.19: Paso 1. Creación de roles.

Unha vez creado o rol e asignados os permisos, deben indicarse os membros que pertencen a dito rol.

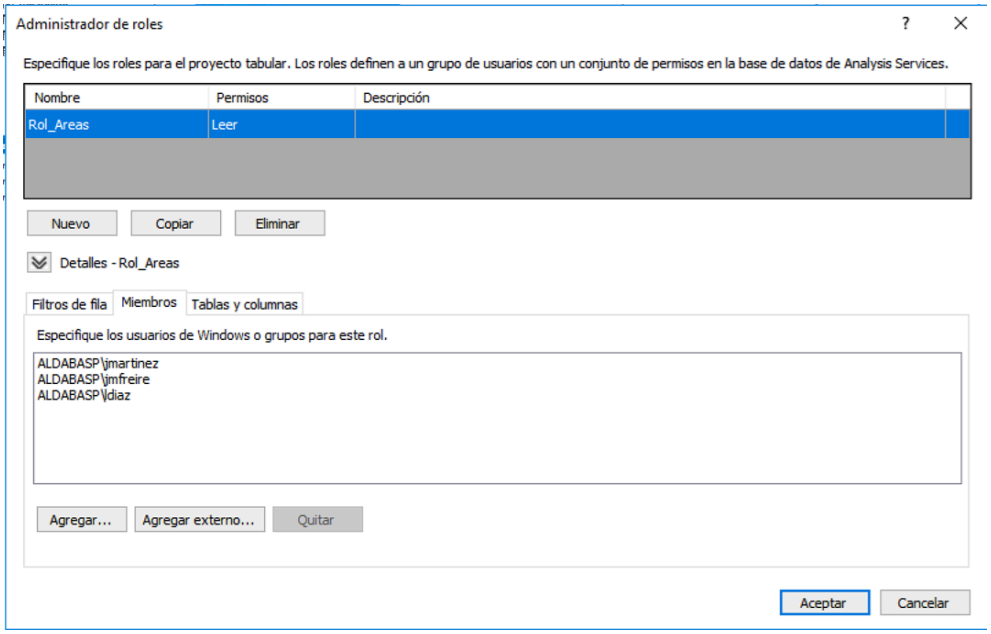

Figura 5.20: Paso 2. Creación de roles.

5.2.8 Procesamento do modelo Tabular

O procesamento do modelo durante o desenvolvemento realízase manualmente e pode realizarse tanto dunha partición, dunha táboa ou do modelo completo.

Tamén é posible procesar o modelo dende o Sql Server Management Studio.

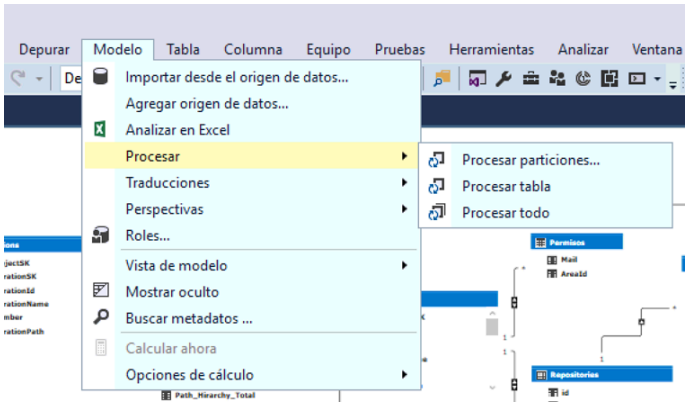

Figura 5.21: Procesamento manual Modelo Tabular.

Tras a creación da versión final do modelo para evitar procesar manualmente, creamos un Job que se executa diariamente [10].

Para crear un job é preciso ter permisos de administrador na BD, e dende o Management Studio, no explorador de obxectos debemos acceder ao Axente de SQL Server. Dentro deste,

atópanse os diferentes traballos creados con anterioridade. Dende aquí pódese crear un novo job que neste proxecto ten o nome de ”Process Model TFG”:

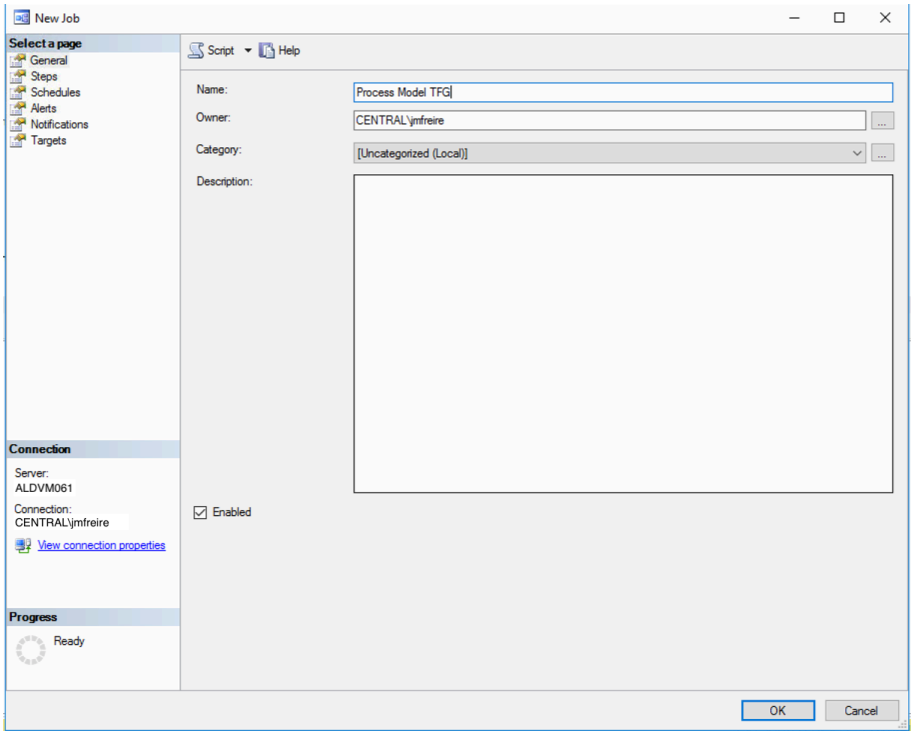

Figura 5.22: Paso 1. Creación do JOB.

Se se desexa recibir unha notificación cando o job non se executa correctamente, pode indicarse que se envíe un correo electrónico.

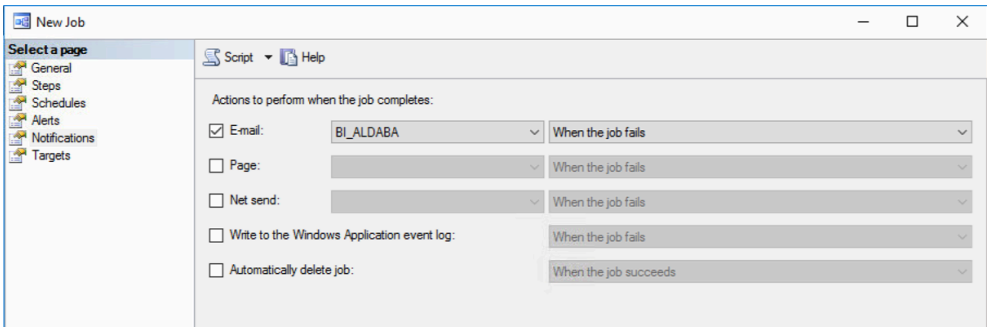

Figura 5.23: Paso 3. Notificación do JOB.

Unha vez creado, debe engadirse un paso (step) no cal se indica un nome para o paso, o tipo (Analysis Services), o servidor e o código a executar. Neste caso o job só contén un paso, pero podería incluír tantos como foran precisos.

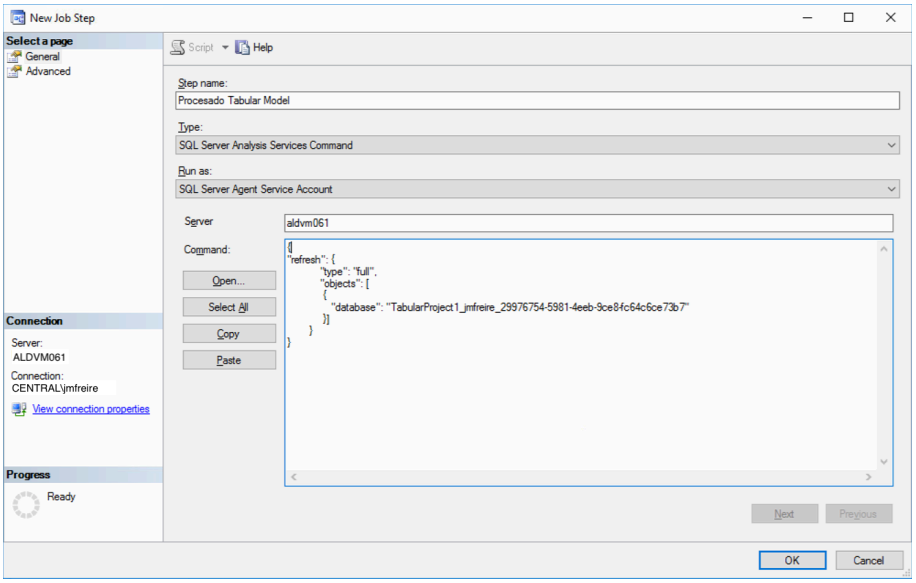

Figura 5.24: Paso 2. Creación step do JOB.

Tamén debe de incluírse unha programación para que o job se execute diariamente. Neste caso indícase que a execución se realice ás 6 da madrugada.

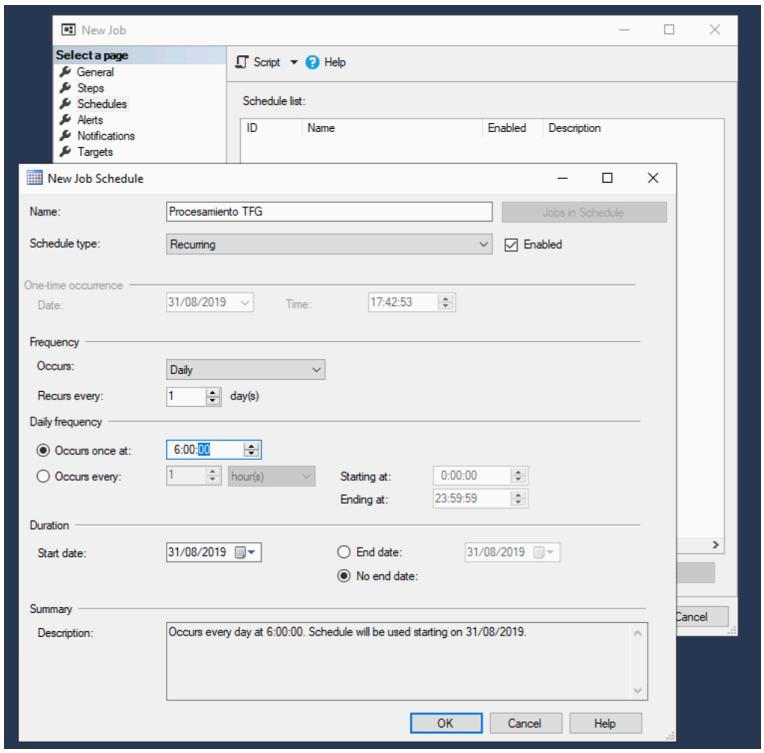

Figura 5.25: Paso 4. Programación do JOB.

Capítulo 6

Cadro de Mando

Neste capítulo explícase a creación do cadro de mando e a publicación deste na web de PowerBI coa correspondente asignación de permisos.

6.1 Integración de Power BI con SSAS

A obtención de datos dende SSAS a Power BI pode realizarse empregando o seguinte compoñente dipoñible na ferramenta:

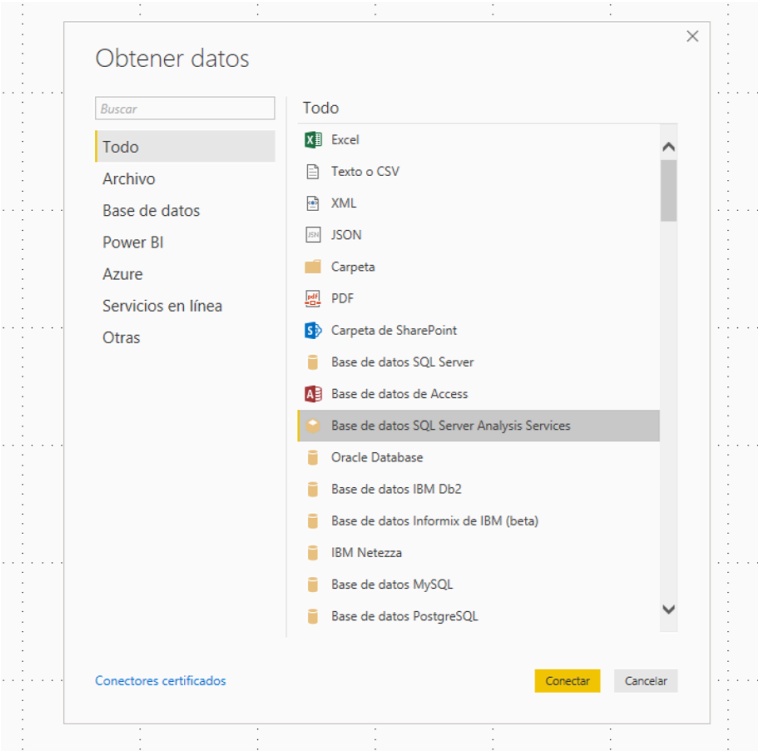

Figura 6.1: Obtención de datos dende SSAS.

Esta forma de obtención de datos non nos permitiría publicar os informes na web nin polo tanto a creación dunha área de traballo onde diferentes usuarios da organización poidan consultar os datos analizados.

Neste proxecto, obtamos pola instalación dunha porta de enlace de datos local no servidor. Deste xeito, podemos ter acceso aos datos locais dende a nube e con diferentes usuarios da organización.

Para a creación da porta de enlace, debemos descargar o executable dende a web de Power-BI e posteriormente instalalo no servidor onde temos os datos almacenados seguindo os pasos que se indican no Anexo A.

6.2 Creación do conxunto de datos

Unha vez instalada e configurada a porta de enlace no servidor local, temos que crear un conxunto de datos. Para a creación deste, debemos acceder ao gateway dende a web de PowerBI tal e como se mostra na seguinte imaxe:

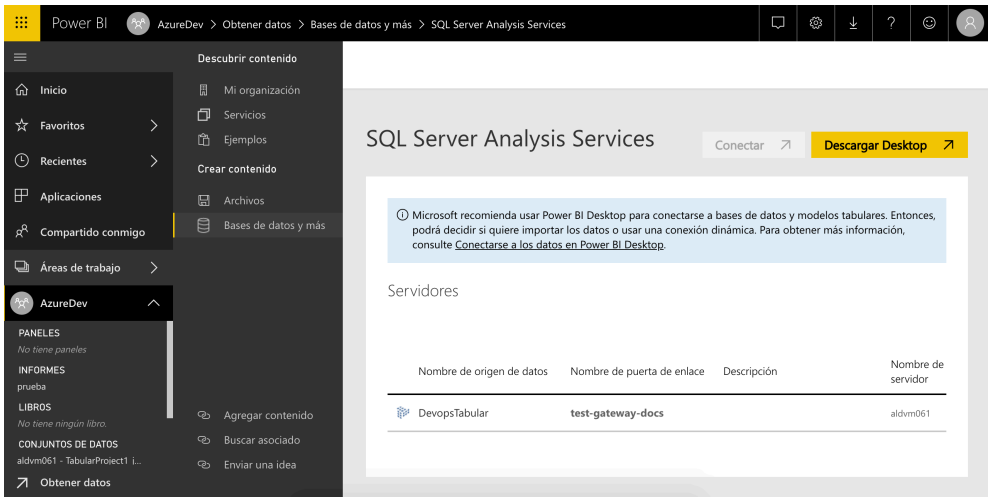

Figura 6.2: Creación conxunto de datos.

6.3 Creación do cadro de mando

Tras a creación do conxunto de datos, podemos crear os informes requeridos dende a aplicación de escritorio de Power BI.

Ao abrir dita aplicación, debemos seleccionar as opcións de *Obter Datos* > *Power BI* > *Conxunto de datos de Power BI.*

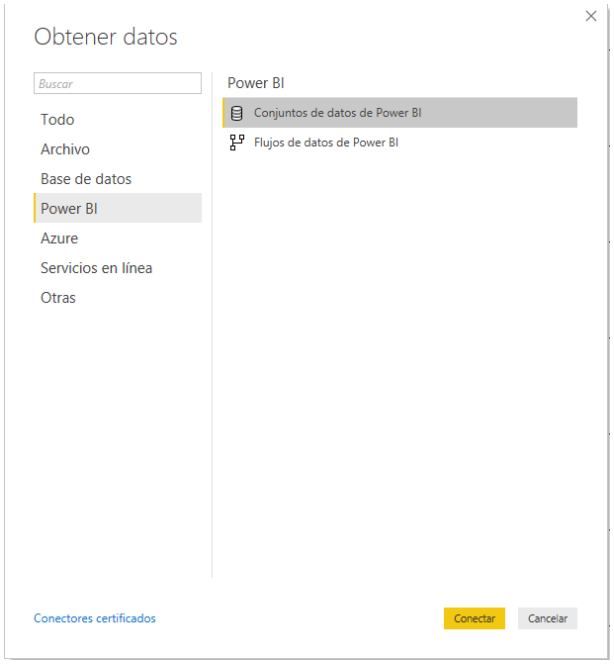

Figura 6.3: Obtención de datos dende a Porta de Enlace.

Tras dita navegación, debemos seleccionar a orixe de datos creada previamente dende a Web. Unha vez obtidas as táboas do modelo tabular, podemos comezar coa creación dos informes.

Deste xeito, os novos informes a crear empregarán como orixe de datos a porta de enlace, polo que poderán ser publicados na web de PowerBI e compartidos con distintos usuarios da organización.Ditos informes serán accesibles por todos os usuarios que teñan permisos a área onde estén publicados.

Unha vez conectados á porta de enlace, obtemos as táboas do modelo, polo que se pode comezar coa creación do cadro de mando.

Para a construcción deste, empregamos as diferentes visualizacións das que dispón PowerBI arrastrándoas á parte esquerda e engadíndolles os diferentes campos das táboas. Se ó incorporar os campos ós compoñentes se produce algún erro, é debido a que as relacións creadas no modelo non permiten esa combinación.

Dentro do menú de visualizacións, además de engadir diferentes tipos de compoñentes, podemos darlles formato, incluír título...

Tamén faremos uso da vista de filtros tanto para os compoñentes como para as distintas pestanas do cadro de mando.

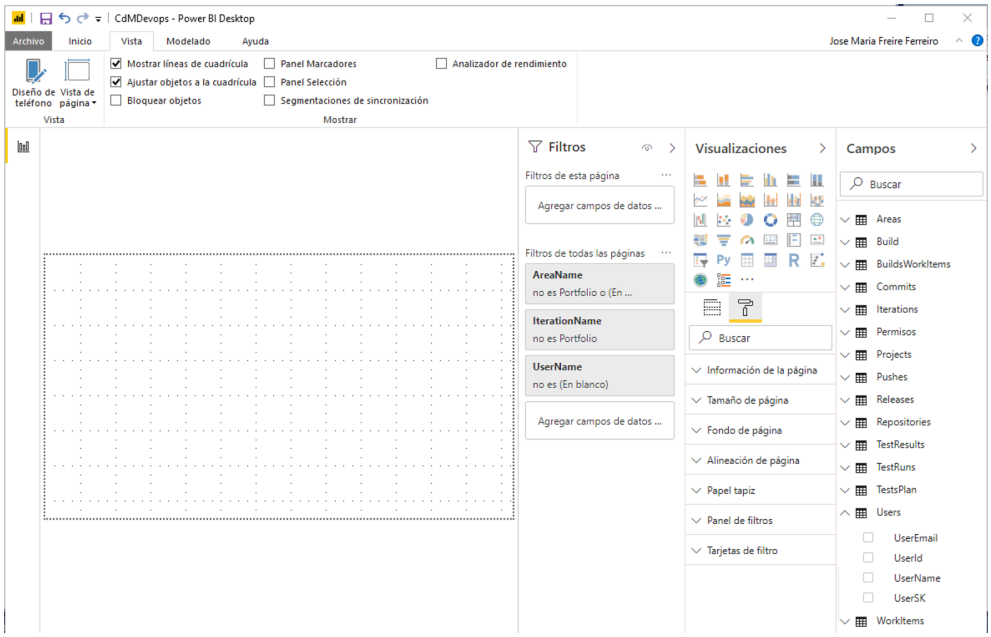

Figura 6.4: Creación do Cadro de Mando.

O cadro de mando xerado está composto por diferentes pestanas:

- **Pestana 1:**

  Mostra unha serie de indicadores acerca de todos os proxectos da empresa. Nestes indicadores podemos observar datos de especial relevancia, como o número de releases, de equipos de traballo, de historias de usuario, a data de obtención dos datos...

  Ademáis dos indicadores, engádense no cadro de mando 3 gráficos:

  - Gráfico de aneis, onde se indica a porcentaxe dos diferentes resultados dos test runs.
  - Gráfico de barras apiladas, onde para cada área de traballo, podemos observar as distintas porcentaxes de tipos de workitems.
  - Gráfico de barras apiladas que mostra para o conxuntos de proxectos a porcentaxe de cada un dos estados para os diferentes tipos de workitems.

Figura 6.5: 1. Cadro de Mando.

- **Pestana 2:**

  Esta pestana inclúe dous filtros en forma de desplegable, onde podemos filtrar por equipo e por iteración.

  A parte esquerda contén un gráfico de barras xunto cunha táboa, onde para cada usuario do equipo se mostran os días de resolución das historias de usuario. Na táboa temos máis detalles acerca das historias de usuario.

  Na parte superior dereita creáronse dous indicadores onde podemos observar o número de tests e cantos deles foron pasados con éxito.

  Xunto aos indicadores, incluíuse un gráfico de aneis que nos indica unha porcentaxe acerca do estado das historias de usuario.

  Esta pestana do cadro de mando tamén contén un gráfico de barras indicando o estádo das tarefas de cada un dos usuarios e unha táboa con máis datos acerca destas.

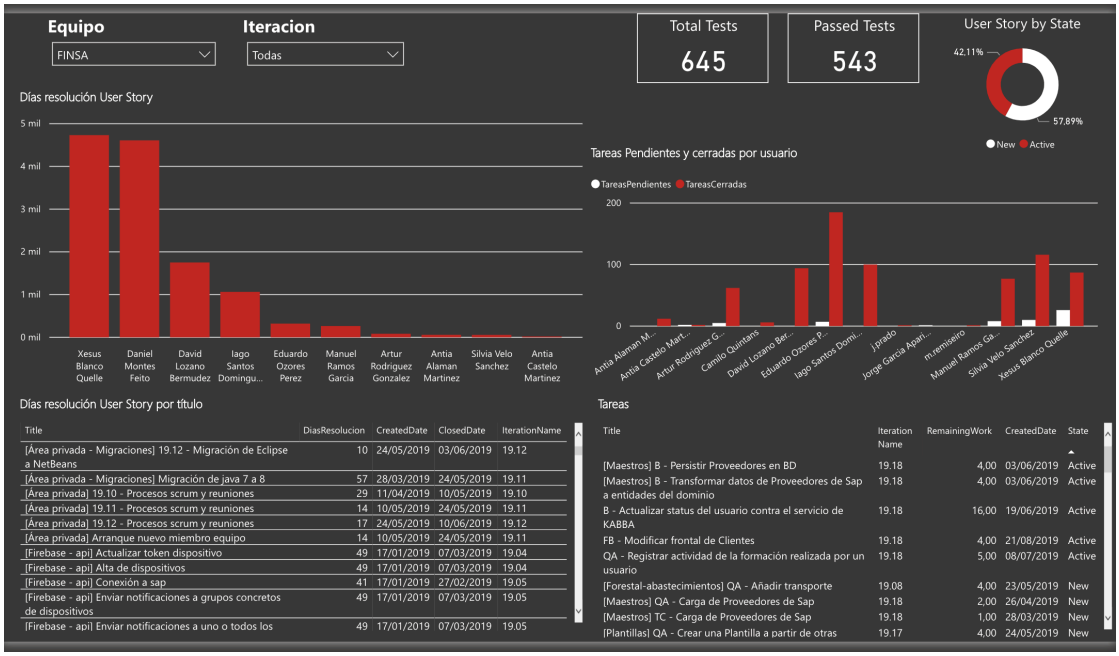

Figura 6.6: 2. Cadro de Mando.

- **Pestana 3:**

  De igual xeito que na anterior, podemos filtrar por tanto por equipo como polas distintas iteracións.

  Esta pestana contén unha táboa xerárquica cos diferentes workitems, pola cal é posible navegar dende os niveis máis altos (Epics) aos de máis baixo nivel.

  Na parte dereita engadíronse 3 indicadores que nos mostran o número de pushes, commits e bugs pendentes de resolver.

  Debaixo dos indicadores mostramos dúas táboas, onde a primeira delas nos indica para cada iteración o tempo restante e o estado de cada unha delas. Idealmente o tempo restante para as iteracións finalizadas debería de ser 0, pero algún dos workitems posiblemente non foi actualizado tras cerrar a iteración.

  Na segunda das táboas comentadas, indicamos as distintas tarefas, o estado,o tempo restante de cada unha delas, a fecha de creación e a iteración a que prtencen.

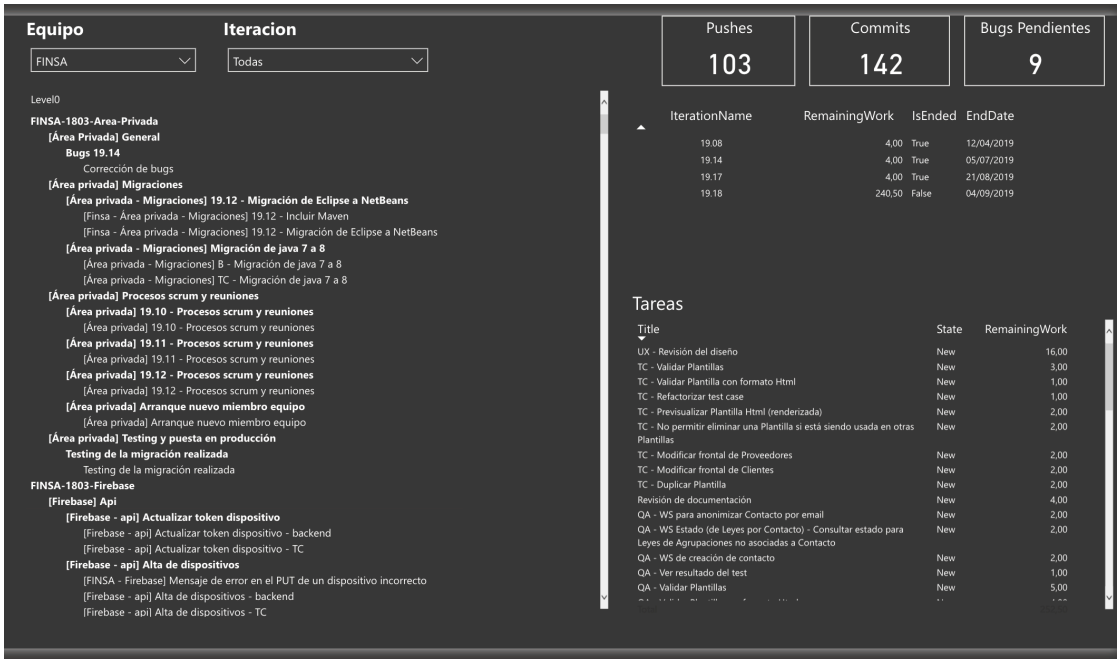

Figura 6.7: 3. Cadro de Mando.

- **Pestana 4:**

  Inclúe información relacionada coas Buidls, onde podemos filtrar os datos por equipo de traballo.

  Esta pestana contén dous indicadores onde o primeiro mostra o número de builds total e o segundo indica en cantas delas o resultado foi fallido.

  Gráficamente, móstrase a duración (en minutos) por día das builds.

  En forma de táboa obtéñense os detalles das diferentes builds, como son o nome, o estado, a prioridade...

  Tamén inclúe unha segunda táboa dode se mostran os datos dos workitems correspondentes.

  Tanto o comportamento desta pestana como o das anteriores é dinámico, e dicir, se pulsamos por exemplo sobre unha das barras do gráfico, todos os compoñentes do panel filtraranse polo día seleccionado.

Figura 6.8: 4. Cadro de Mando.

6.4 Creación da área de traballo e asignación de permisos

A creación dunha área de traballo realízase pulsando o botón 'Crear área de traballo' que se indica na imaxe superior. Posteriormente, debemos de indicar un nome para dita área e opcionalmente unha descrición.

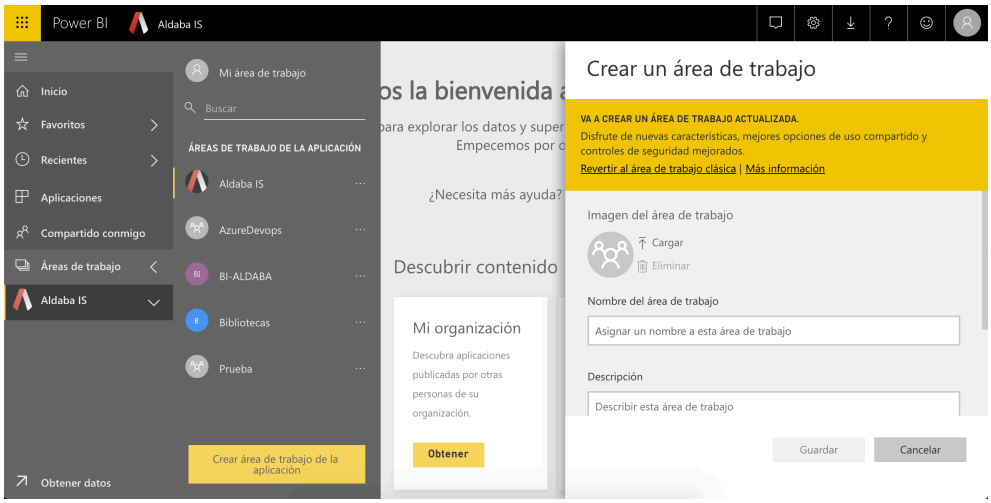

Figura 6.9: Creación da Área de traballo.

Unha vez creada a área de traballo, debemos de incluír aos usuarios que desexamos darlles permisos de acceso.

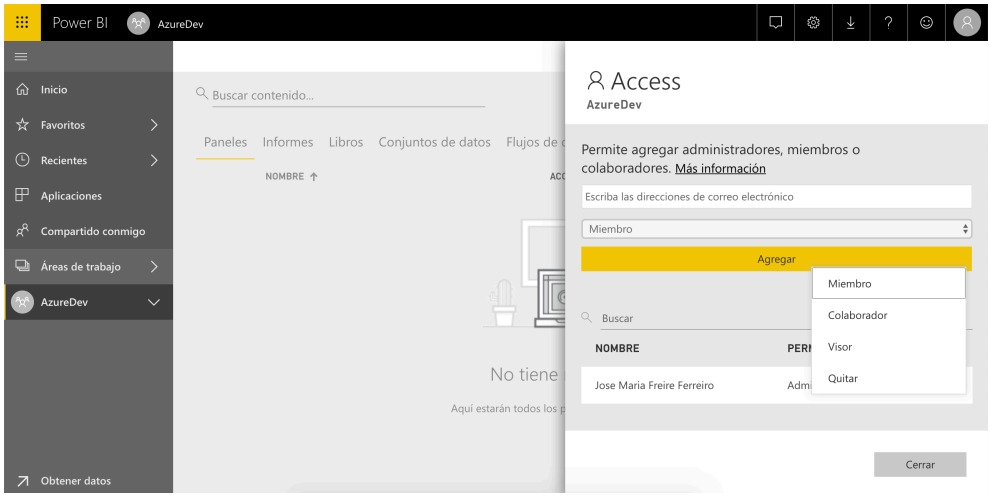

Figura 6.10: Agregación de usuarios á Área de traballo.

Dende esta ventana podemos asignar varios tipos de permisos: Membro, Colaborador ou Visor.

Durante o desenvolvemento dos informes, podemos publicalos nunha área de traballo persoal dentro da nosa conta de Power BI.

Para publicar, dende a aplicación de escritorio de PowerBI, debemos ter iniciada sesión coa nosa conta e posteriormente pulsar o botón de Publicar e seleccionar a área de traballo correspondente, ben sexa persoal (en fases iniciais da creación de informes) ou compartida co resto de usuarios cos que desexamos compartir os informes creados.

Capítulo 7

Incidencias, conclusións e traballo futuro

DURANTE este capítulo analízanse as incidencias surxidas durante o desenvolvemento do traballo, explicando os motivos de cada unha delas e xustificando o seu trato.

Tamén comentaremos as posibles melloras, alternativas e traballo futuro. Finalmente, remataremos coas conclusións obtidas tras a finalización do proxecto

7.1 Incidencias

7.1.1 Versión instalada da instancia tabular

Tras a instalación da instancia tabular, á hora de crear o modelo, decatamonos de que a versión de Análisys Service instalada debería de ser unha posterior, xa que o compoñente de oData non estaba dispoñible, polo que foi entón preciso actualizar a 2017.

7.1.2 Obtención de datos

Un dos principais problemas surxiu á hora de importar os datos ao modelo tabular dende Azure Devops, xa que nun primeiro momento os datos obtíñanse mediante o protocolo oData, pero decatámonos de que faltaban moitos datos como era por exemplo o caso dos TestCase. Tras un proceso de investigación, decidimos crear un servizo en C#, que mediante API Rest, podemos obter estes datos e fusionalos cos obtidos mediante oData.

Esta investigación e creación do servizo, retrasou considerablemente a creación do modelo e as melloras do cadro de mando tanto funcionais como de deseño.

7.1.3 Importación de datos e publicación do informe

Nunha primera fase de creación do cadro de mando, tentouse realizar unha importación do modelo tabular a PowerBI, pero isto provocaba un cross join de todas as táboas do modelo, polo que producía un erro de memoria. Tras este problema, tomouse a decisión de non importar os datos ao cadro de mando, senón de realizar unha conexión en vivo co modelo.

Unha vez creada unha pequena versión do cadro de mando, o problema surxiu ao intentar publicalo, xa que era necesaria a instalación do Gateway no servidor.

7.2 Conclusións

Tras a finalización do traballo, os resultados obtidos son similares as necesidades previas á realización do proxecto por parte da organización.

Conseguiuse acadar un sistema, dende o cal os membros da organización poden consultar os datos relativos aos proxectos que se están a desenvolver.

Grazas a realización deste traballo, tamén foi posible detectar que o uso de AzureDevops en moitas ocasións non é o correcto por parte dos usuarios. Agora simplemente accedendo ao Cadro de Mando creado, podemos detectar grupos de traballo ou usuarios con sobrecarga, tarefas que levan moito tempo sen ser resoltos, duración das builds etc..

A aprendizaxe ao longo do desenvolvemento do proxecto foi continua, sobre todo en cuestións técnicas, como é o caso da linguaxe C# ou DAX, a creación de modelos tabulares e o emprego de ferramentas como Visual Studio ou PowerBI, e por outro en cuestións máis teóricas como é o modelado.

Este proxecto supuxo tamén moitas horas de investigación que quizais non se vexan directamente reflectidas na memoria final, sobre todo para tomar as decisión to tipo de modelo a crear e a forma de obtención dos datos dende AzureDevops.

Toda esta aprendizaxe, foi aplicada en paralelo, para o deselvolvemento dun proxecto similar a este, pero de maior magnitude, para un cliente. Este proxecto é o primeiro que se realiza na organización destas características.

Os problemas de implementación que foron xurdindo foron superados pouco a pouco para cumprir os requerimentos, e mesmo houbo que refacer parte do modelo en máis dunha ocasión.

7.3 Traballo futuro

Tras cumprir os obxectivos iniciais é preciso pensar en futuras liñas de traballo que poden ser clasificadas nos seguintes tipos:

- A principal, correspóndese co mantemento do sistema, xa que é preciso que os datos mostrados aos diferentes usuarios con acceso á plataforma sexan correctos e estén actualizados.

- O seguinte, sería realizar traballos a curto prazo, como pode ser a mellora do deseño do cadro do mando ou a incorporación e modificación de medidas, cuadrículas e gráficos que vaian considerándose interesantes ao longo do tempo. Outra parte importante, pero non requerida para esta primeira fase, é tratar a seguridade dos informes a partir dos usuarios activos de Windows.

- O último tipo componse de ideas a máis longo prazo, como pode ser a incorporación ao modelo de datos obtidos dende outras plataformas e o fusionamento destes datos para a creación de novos informes que poidan ser útiles para a organización, como pode ser a plataforma de fichaxes.

  Outro aspecto a ter en conta a longo prazo é a substitución do servizo por un ETL dende o cal se obteñan os datos de Azure Devops. Facendo uso do ETL, poderíamos deste xeito gardar nunha BD o estado dos datos en diferentes instantes de tempo para poder así facer comparacións.

Apéndices

Apéndice A

Instalación do software de desenvolvemento.

A continuación imos mostrar os pasos a seguir para instalar o software requerido para o desenvolvemento do proxecto. Instalamos no servidor facilitado por parte da empresa o servizo creado para a descarga dos datos, Análysis Services en modo tabular e a porta de enlace. No equipo local instalamos PowerBI Desktop e Visual Studio co modelo de Business Intelligence (SSTD).

A.1 JSONDownloader

Para a instalación do servizo creado, dende a liña de comandos de Windows debemos de executar:

```
1        installutil JSONDownloader.exe
2
```

A.2 PowerBI

Para a instalación de Power BI Desktop accedemos á url de PowerBI de onde descargamos o executable para a instalación local [11].

É precisa unha conta Pro para poder compartir a visualización de informes por diferentes usuarios.

A.3 Analysis Sevices

Xa que o servidor facilitado pola organización, non dispoñía de SQL Server Análysis Services, foi precisa a súa instalación.

Como se mostra nas seguintes imaxes, debemos de crear unha nova instalación de SQL Server 2016.

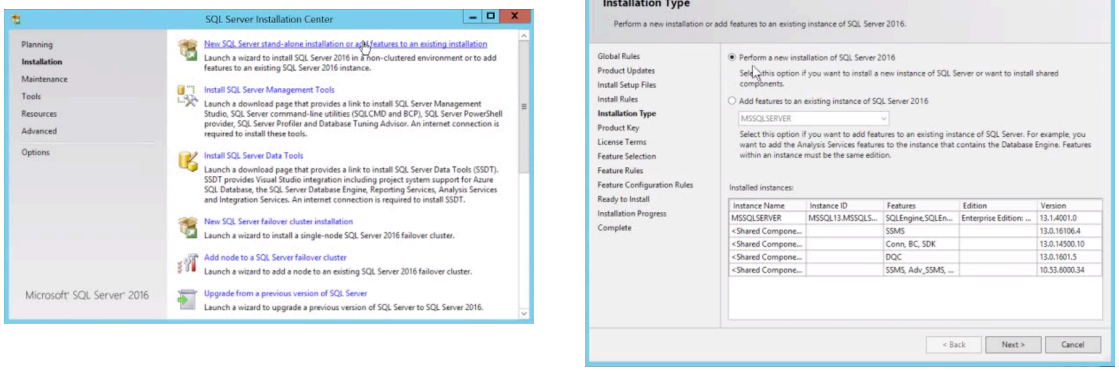

Figura A.1: Instalación Analysis Services.

Posteriormente debemos de seleccionar Analysis Services e como modelo a instalar o tabular:

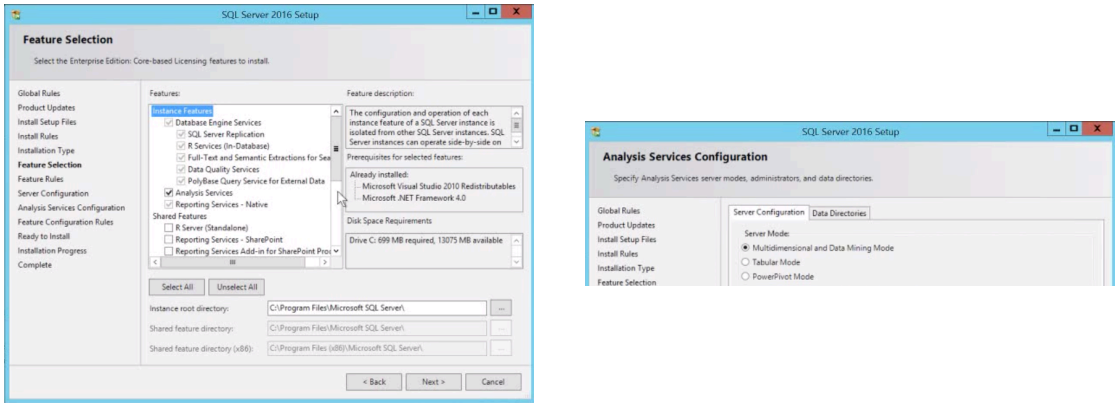

Figura A.2: Instalación Analysis Services.

A.4 Instalación e configuración da porta de enlace

Para a instalación da porta de enlace, debemos de acceder á url [12], descargar o executable e executalo. Unha vez executado debemos de seleccionar o tipo de porta de enlace a instalar, e a ruta onde será instalada:

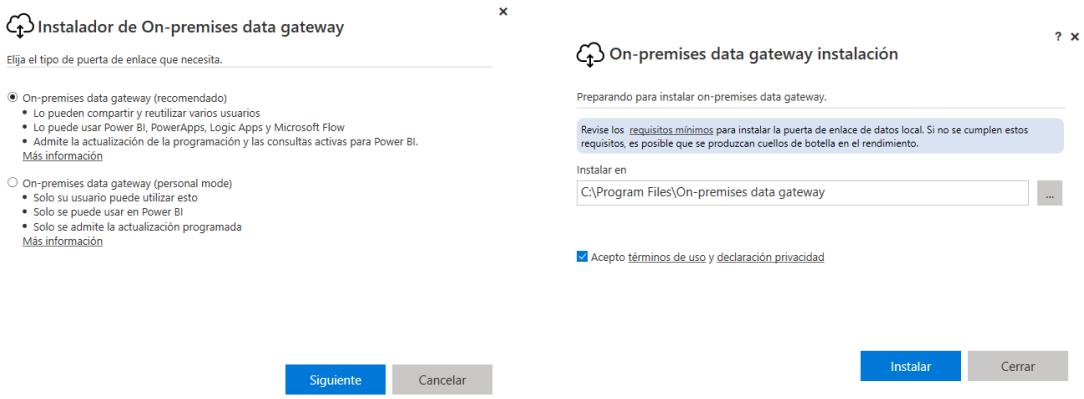

Figura A.3: Instalación Gateway.

Posteriormente, debemos de indicar a nosa conta de Power BI, e neste caso, tamén indicamos que se trata dunha nova porta de enlace.

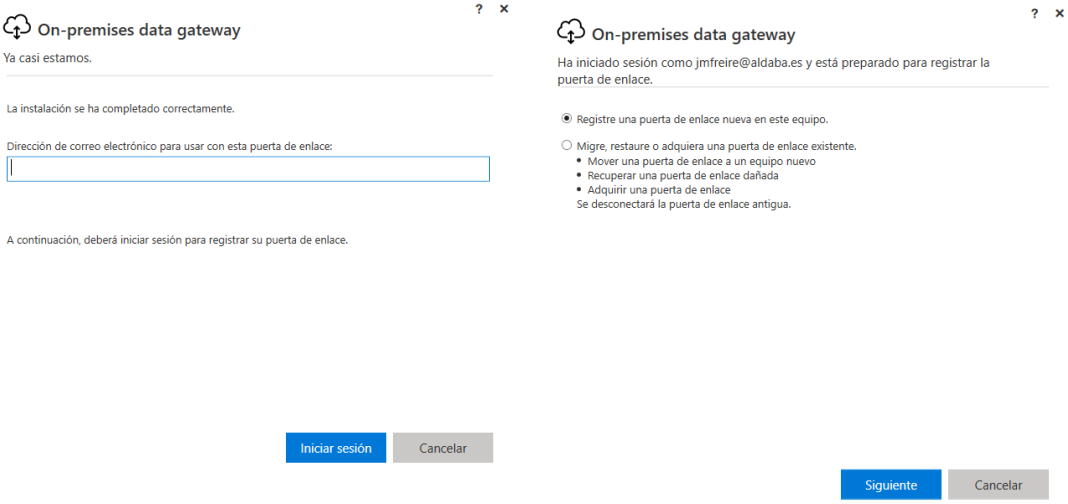

Figura A.4: Rexistro do Gateway.

Unha vez instalada a Porta de Enlace, podemos ver o estado desta, reiniciala, cambiar a conta e outras funcionalidades.

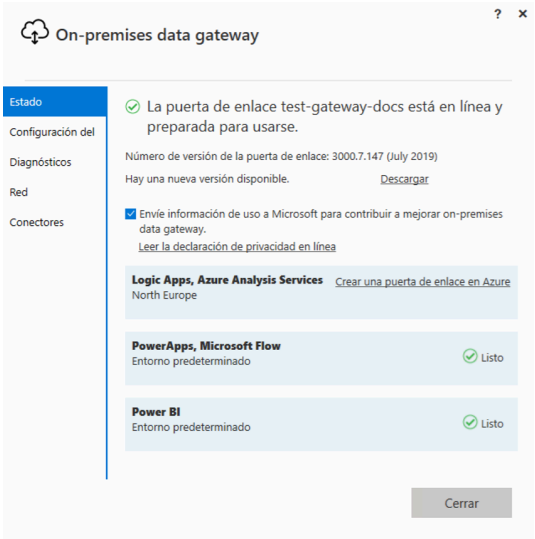

Figura A.5: Estado do Gateway.

A.5 Visual Studio

A instalación de Visual Studio realízase tras a descarga do executable dende a url (neste caso a versión 2015) [13].

Tras a instalación de Visual Studio, é preciso instalar SSDT (SQL Server Data Tools) para a creación do modelo tabular.

Apéndice B

Planificación e estimación de custos

Neste apartado, realízase unha pequena descrición sobre cada unha das fases do proxecto, unha estimación do tempo e do custo do proxecto antes de comezar o desenvolvemento e tras a presentación do diagrama de Gantt.

B.1 Planificación de tarefas

As etapas que comprenden a planificación datan dende finais de abril do ano 2018, ata a data de entrega deste proxecto.

Dentro deste rango de datas pódense diferenciar 2 etapas por diversos motivos (aclaración de requisitos e tecnoloxías a empregar en consonancia coa organización, migración de proxectos a Azure Devops, alta carga de traballo noutros proxectos...).

A continuación móstrase con maior detalle cada unhas das fases do proxecto.

B.1.1 Análise do problema a resolver

Comprende dende finais de Abril 2018 ata mediados de Maio, onde se realizou unha primeira toma de contacto. As tarefas realizadas son:

- Recopilación e análise da información obtida.
- Análise de viabilidade.
- Alternativas comerciais.
- Estudo de ferramentas e tecnoloxías a empregar tendo en conta a calidade, a facilidade de implantación ou a interoberabilidade.
- Instalación do software.
- Formación nas ferramentas e nas tecnoloxías.

B.1.2 Desenvolvemento

Antes de comezar coa implementación, realízase unha análise da arquitectura do sistema e defínense as diferentes iteracións nas que se divide o desenvolvemento.

Todas as iteracións constan de catro tarefas: Análise, Deseño, Implementación e Probas.

A continuación realízase unha pequena explicación acerca das diferentes tarefas que están contidas en cada unha destas iteracións. Despois de finalizar cada iteración, disporase dunha versión funcional do programa.

Iteración 1:

- Análise: Análise da obtención de datos e a estrutura do modelo tabular.
- Deseño: Deseño xeral do modelo tabular.
- Implementación do modelo tabular: Creación da estrutura do modelo tabular cun exemplo de proba.
- Implementación do cadro de mando: Creación dun exemplo de cadro de mando que ten como orixe o modelo.
- Probas: Validación do cadro de mando cos datos de proba.

Unha vez finalizada esta iteración, decídese agardar un tempo xa que os proxectos non están migrados a Azure Devops. Ata este momento existía un proxecto de proba para poder comerzar o desenvolvemento.

Iteración 2:

Marca o comezo da segunda etapa comentada ao inicio deste apartado.

Xorde a necesidade de crear un servizo para a obtención do resto de datos de Azure Devops que non é posible obter mediante o protocolo oData.

- Análise: Análise sobre as táboas novas a engadir e a maneira de obtelas. Nesta iteración xa dispoñemos de datos reais.
- Deseño: Deseño do modelo e prototipo do cadro de mando.
- Implementación do servizo: Creación do servizo con API Rest.
- Implementación do modelo tabular: Incorporación das novas taboas ao modelo.
- Implementación do cadro de mando: Modificación do cadro de mando (agora con datos reais)

- Probas: Validación de datos.

Iteración 3:

- Análise: Análise das medidas/columnas calculadas e novas táboas a engadir.
- Deseño: Redeseño do modelo tabular e do cadro de mando.
- Implementación do servizo: Incorporación das novas funcionalidades ao servizo.
- Implementación do modelo tabular: Modificación das relacións, incorporación das táboas e creación das medidas e columnas calculadas.
- Creación da porta de enlace: Instalación e configuración da porta de enlace.
- Implementación do cadro de mando: Creación do cadro de mando coa porta de enlace como orixe de datos.
- Probas: Validación dos datos.

Iteración 4:

- Análise: Estudo exhaustivo de todos os aspectos restantes para o fin do proxecto.
- Deseño: Modificación do deseño do cadro de mando e incorporación das cores corporativas da organización.
- Implementación do modelo tabular: Novas medidas e roles de seguridade. Tamén se engade o job de procesamento.
- Implementación do cadro de mando según o deseño. Engádese a área de traballo e asignanse os permisos.
- Probas: Validación de datos e dos filtros do cadro mando xunto co correcto funcionamento do servizo.

B.1.3 Documentación

Elabórase un documento detallando o traballo realizado durante as fases comentadas anteriormente. Este traballo é levado a cabo conxuntamente entre o xefe de proxecto, o analista e o programador.

B.2 Riscos

- Descoñecemento das tecnoloxías a empregar para realizar o proxecto.
- Inexperiencia á hora de estimar proxectos.
- Actualización da versión API REST.
- Cambios na estrutura dos JSON por parte de Azure Devops.

B.3 Estimación de custos

Para o cálculo do custo total debemos ter en conta os seguintes tipos de gastos:

- **Recursos hardware:** Empregouse un ordenador sobremesa con dous monitores valorado en 700€.
- **Recursos software:** Licenza de PowerBI Pro para cada usuario que se desexa darlle acceso á area de traballo. O prezo da licenza é de 120€ por cada usuario ó ano. Para calcular o custo para este proxecto, temos en conta o custo de 1 ano para 4 usuarios, polo que sería de 480€.
- **Recursos humanos:** Para o cálculo sobre o custo que supón o persoal para a realización deste proxecto, debemos de ter en conta os distintos salarios:

Cargo	**€/h**
(XP) Xefe Proxecto	65€
(A) Analista	50€
(P) Programador	40€
(TS) Técnico Sistemas	40€

Táboa B.1: Táboa de salarios

Na seguinte táboa móstranse os custos por cada tarefa dependendo da duración eo cargo do persoal que a desempeña:

	Duración	**Custo**	**Recurso**
Análise do problema a resolver	**58 h**	**2.670€**	
Recopilación e análise da información	16 h	920 €	XP / A
Análise da viabilidade	4 h	230 €	XP / A
Estudo de ferramentas e tecnoloxías	10 h	400 €	P
Instalación do software	8 h	320 €	TS
Formación	20 h	800 €	P
Desenvolvemento	**256 h**	**10.625€**	
Arquitectura xeral	16 h	920 €	XP / A
Definición das iteracións	8 h	460 €	XP / A
Iteración 1	**60 h**	**2.490€**	
Análise	6 h	300 €	A
Deseño	6 h	270 €	A / P
Implementación Modelo Tabular	24 h	960 €	P
Implementación CdM	16 h	640 €	P
Probas	8 h	320 €	P
Iteración 2	**72 h**	**2.650€**	
Análise	6 h	300 €	A
Deseño	6 h	270 €	A / P
Implementación do servizo	40 h	1.600 €	P
Implementación Modelo Tabular	8 h	320 €	P
Implementación CdM	8 h	320 €	P
Probas	4 h	160 €	P
Iteración 3	**40 h**	**1.685€**	
Análise	3 h	150 €	A
Deseño	2 h	135 €	A / P
Implementación do servizo	16 h	640 €	P
Implementación Modelo Tabular	8 h	320 €	P
Implementación CdM	8 h	320 €	P
Probas	3 h	120 €	P
Iteración 4	**60 h**	**2.420€**	
Análise	2 h	100 €	A
Deseño	16 h	720 €	A / P

Continúa na páxina seguinte.

	Duración	**Custo**	**Recurso**
Implementación Modelo Tabular	8 h	320 €	P
Implementación CdM	24 h	960 €	P
Probas	8 h	320 €	P
Documentación	**90 h**	**4.650 €**	
Creación da Memoria	90 h	4.650 €	XP /A / P
Total	**404 h**	**17.945 €**	

Táboa B.2: Táboa de tarefas

Tras o cálculo individual de todos os gastos, podemos finalmente obter o custo total do proxecto:

Tipo de Recurso	**Custo**
Humano	17.945 €
Software	480 €
Hardware	700 €
Custo total do proxecto	**19.125 €**

Táboa B.3: Táboa de custos

B.4 Diagrama de Gantt

A continuación móstrase o diagrama de Gantt dividido nas dúas fases comentadas anteriormente para a súa mellor visualización.

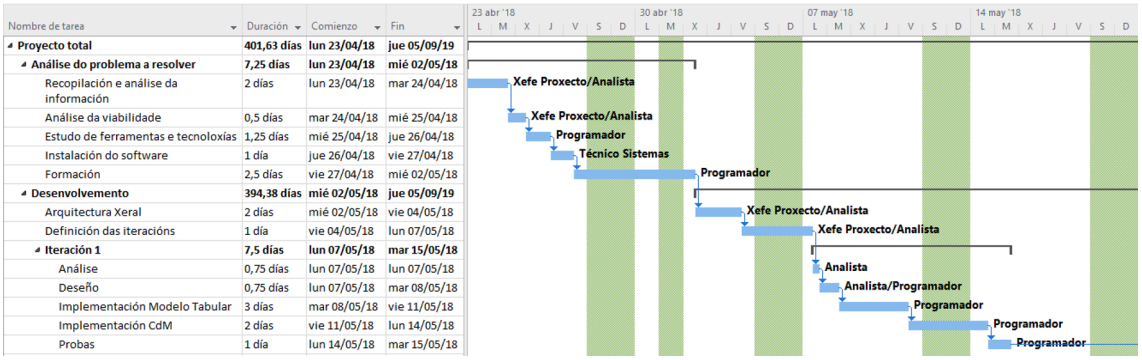

Figura B.1: Fase 1. Diagrama de Gantt.

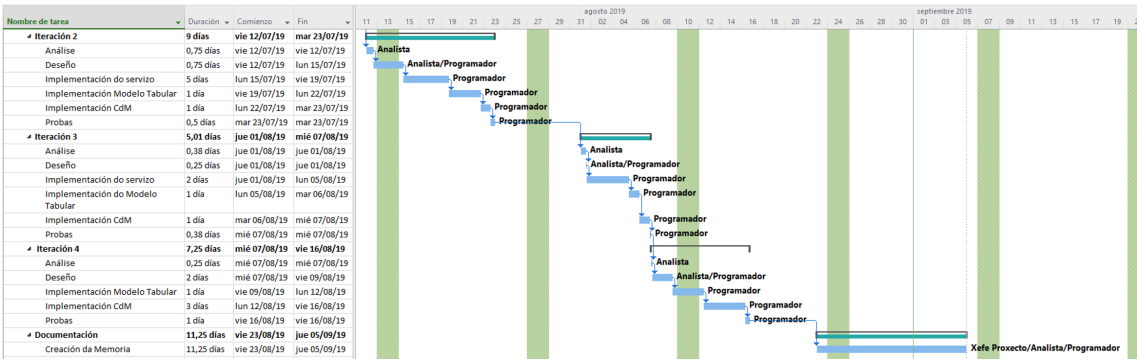

Figura B.2: Fase 2. Diagrama de Gantt.

Apéndice C

Contido do CD

O CD entregado, tal e como establece o regulamento, contén o seguinte arquivo:

- Copia electrónica da memoria en formato PDF:

  FreireFerreiro_JoseMaría_TFG_2019.pdf

Bibliografía

[1] R. B. G. Mercado., "Data warehouse." 2019, http://www.rberny.com/blog/data-warehouse/.

[2] B. kumar Singh, "Microsoft business intelligence," http://bageshkumarbagi-msbi.blogspot.com/2014/12/defining-ssas-data-warehouse.html.

[3] Mquantin, "Cube olap et opérations (drill-up, drill-down, dicing, roll-up, slicing, pivot)." 2017, https://commons.wikimedia.org/wiki/File:OLAPcube.png.

[4] "2019 gartner magic quadrant for analytics and business intelligence platforms." 2019, https://info.microsoft.com/ww-landing-gartner-mq-bi-analytics-2019.html?LCID=EN-US.

[5] S. Macias., "Azure devops: Creando el proyecto, equipos de trabajo y repositorios git." 2019, https://enmilocalfunciona.io/azure-devops-creando-proyecto-equipos-de-trabajo-git/.

[6] lmblanco., "Introducción a azure devopsmodelos tabulares en sql server 2012 analysis services." 2015, https://geeks.ms/lmblanco/2015/04/27/modelos-tabulares-en-sql-server-2012-analysis-services/.

[7] J. Cool., "Introducción a azure devops." 2018, https://azure.microsoft.com/es-mx/blog/introducing-azure-devops/.

[8] "Qué es una puerta de enlace de datos local?." https://docs.microsoft.com/es-es/power-bi/service-gateway-onprem.

[9] "Azure devops services rest api reference." 2016, https://docs.microsoft.com/en-us/rest/api/azure/devops/?view=azure-devops-rest-5.1.

[10] "Cómo automatizar el procesamiento de modelo tabular ssas en sql server 2016." https://www.sqlshack.com/es/como-automatizar-el-procesamiento-de-modelo-tabular-ssas-en-sql-server-2016/.

[11] "Vaya de la obtención de datos a la información y a la acción con power bi desktop." https://powerbi.microsoft.com/es-es/desktop/.

[12] "Mantenga sus paneles e informes al día con los orígenes de datos locales," https://powerbi.microsoft.com/es-es/gateway/.

[13] "Visual studio," https://visualstudio.microsoft.com/es/vs/older-downloads/.