Dashboards para democratização dos dados dos Building Information Models

https://doi.org/10.21814/uminho.ed.142.27

José Mota[1], **Miguel Pires**[1], **Pedro Carneiro**[1]

[1] *TOPBIM, Braga, Portugal*

Resumo

Um desafio associado ao desenvolvimento de modelos de informação de projecto, aquando da implementação do BIM, é a criação de dados relevantes que acabam por não ser utilizados por terceiros. Isto sucede porque, generalizadamente, quem acede à informação dos modelos o faz exclusiva ou maioritariamente através da documentação de projecto exportada e, motivado pelas carências de software, conhecimento técnico ou posse dos modelos para acesso a informação adicional, à omissão de informação nas peças de projecto, corresponde o seu desconhecimento e consequente não utilização.

Uma das soluções encontradas pelo Grupo Casais foi a implementação de *dashboards* BIM, ou seja, painéis gráficos de consulta interativa das bases de dados dos modelos, organizadas por aplicação, ou seja, conforme o perfil de utilizador a que se destinam, através do *Microsoft PowerBI*. A sua intuitividade dispensa uma capacitação técnica profunda para consulta e interação com os dados. Desta forma, estabelece-se uma relação entre as bases de dados dos modelos e as equipas não projetistas, sem intermediário, para que a consulta possa ser totalmente *ad hoc*, filtrável, dinâmica e adaptável à solicitação do utilizador. O acesso via browser permite também a consulta em qualquer dispositivo, fixo ou móvel.

Neste artigo, é descrita em profundidade a investigação e o desenvolvimento destes *dashboards* através do piloto elaborado para o perfil de técnico orçamentista, incluindo as diferentes abordagens tentadas e dificuldades ultrapassadas até ao produto final. São também descritos os próximos passos, nomeadamente os perfis de utilizador a desenvolver com a mesma metodologia.

1. O problema – informação BIMplícita

Um problema recorrente que a TOPBIM tem encontrado na sua aplicação da metodologia BIM é o da transmissão de informação para terceiros, nomeadamente quando esses não se encontram no mesmo grau de maturidade BIM. O que sucede é que, apesar da profusa informação geométrica e não geométrica constante dos modelos de informação que são produzidos, se o seu recetor puder apenas aceder a essa informação por intermédio das peças de projecto tradicionais (PPT - peças desenhadas e escritas como texto, mapas, tabelas, etc), surge inadvertidamente uma barreira intransponível na consulta dessa informação: se não for explicitamente constante desses elementos, independentemente de existir ou não no modelo, é invisível e desconhecida pelo recetor. Um exemplo evidente é o caso de todos os elementos do modelo exportados em desenhos, nos quais só é explícita a informação descrita graficamente por identificadores (*tags*). A esta classe de informação existente no modelo, mas a que o recetor não consegue aceder, chamamos de informação "BIMplícita" (no BIM, de forma implícita), à semelhança de um *iceberg* de informação do modelo, em que apenas a parte superficial é visível; grande parte dos dados está oculta. Tentou-se resolver este desafio de como tornar, então, explícita e acessível essa informação a terceiros, independentemente do seu grau de maturidade BIM.

Figura 1
Informação "BIMplícita".

2. A solução – *Dashboards* BIM

Pretendeu-se criar um novo entregável, anexo às tradicionais peças de projecto, no qual o utilizador pudesse interagir diretamente com a informação dos modelos, para além do constante nas primeiras. Para este novo formato possibilitar um acesso mais democrático a essa informação, teria de possuir algumas características indispensáveis:

- não obrigar a formação BIM nem formação de um software específico para a sua utilização (para além de noções básicas para manuseio que se conseguissem transmitir rapidamente);
- entregar a informação de forma intuitiva e facilmente sincronizável com os desenvolvimentos do projecto;
- ser específica para cada perfil de recetor-utilizador dos dados.

O formato dos *dashboards* (ou painéis de bordo) BIM tem o potencial para satisfação destas premissas. Ao longo do artigo, serão explicados os passos dados, as dificuldades encontradas e ultrapassadas e as várias metodologias. Espera-se que este documento possa eventualmente também servir de referência a quem pretenda implementar uma solução análoga no seu portefólio de outputs BIM.

3. Microsoft Power BI e o caso de estudo

O programa Power BI é a opção da Microsoft para a criação de *dashboards* e, enquanto ferramenta consagrada e com extensa documentação e suporte online, emergiu como ideal para a implementação pretendida. Uma grande vantagem é que consegue obter dados a partir de uma enorme variedade de ficheiros, nomeadamente de texto (como *csv*), folhas de cálculo, bases de dados, etc., ou seja, ficheiros já habitualmente exportáveis a partir dos modelos de informação de projecto. Por outro lado, é de utilização intuitiva – componentes gráficos (ou *visuals*) de fácil apreensão – e dinâmica – permite filtração definida no momento pelo utilizador e consulta interativa na adaptação ativa dos *visuals* da mesma página (ex.: clicar numa barra de um gráfico de barras repercute-se automaticamente nos outros *visuals*, salientando nestes apenas a porção respetiva à seleção). Esta interatividade é fundamental para que o utilizador consiga alcançar exatamente a informação que pretende e não apenas a informação pré-determinada pelo emissor (como acontece com os outputs estáticos dos modelos como desenhos, tabelas, etc.). O facto de o programa permitir construir os *dashboards* de raiz, com *visuals* e hipótese de filtração específicos, dá-lhe a versatilidade pretendida para gerar *dashboards* adaptados a cada perfil de utilizador de destino, conforme desígnio inicial. Por fim, a funcionalidade prevista pelo *Power BI* de simplesmente alterar os ficheiros de origem de dados é possibilita à partida que sejam preparados os *dashboards* como templates de fácil atualização e adaptação a diferentes modelos de informação. Sem a possibilidade de *templatização*, a morosidade de preparação do *dashboard* poderia inviabilizar a abordagem.

Satisfeita a escolha pelo *Power BI* para elaboração dos *dashboards*, procedeu-se à seleção de um caso de estudo destinada a um técnico de orçamentação como o piloto a desenvolver.

Para tal, estudou-se várias hipóteses no sentido de auxiliar a tarefa de orçamentação através de um *dashboard* BIM em Power BI como um elemento extra das PPT, nomeadamente peças escritas, peças desenhadas e mapa de trabalhos e quantidades (MTQ), enquanto exportáveis da mesma base de projecto, ou seja, o modelo de informação.

4. Abordagem 1: ficheiro base de dados + Power BI

O *dashboard* pretendido para este perfil de utilizador deveriam conter, no mínimo, uma página geral de receção (identificação do projecto, navegação entre páginas, modelo geral, poucos filtros, informação detalhada dos elementos apenas após seleção). Para além desta, pretendia-se uma página que enaltecesse a relação entre MTQ e modelo, com destaque para artigo/descritivo dos elementos e uma quantidade sumária do artigo conforme os filtros aplicados. À semelhança da anterior, pretendia-se também uma página com mais tipos de filtros para além do próprio MTQ, para que o orçamentista pudesse aceder a informação específica de determinado artigo mediante a sua posição no que toca ao edifício: piso e sector. Por fim, considerou-se relevante apresentar uma página resumo, sem modelo 3D, apenas com os artigos com as quantidades mais representativas e respetiva distribuição por piso, sector, categoria de elemento *Revit*, etc., para uma visão mais geral. Inicialmente, foi ponderada a hipótese de elaboração do *dashboard* apenas com a informação não gráfica do modelo, ou seja, exportando diretamente a base de dados (ficheiro *.db*) a partir do *Autodesk Revit* e importando-a no *Power BI*. Contudo, apesar da abordagem poder fazer sentido em termos abstratos para aceder à informação de projecto, não é adequada à função pretendida. Senão, vejamos: num processo tradicional, em que as quantidades de um projecto sejam aferidas através de medição em *CAD* (*polylines*, *hatches*, etc), o que beneficia o técnico orçamentista na sua tarefa é a rastreabilidade visual dos artigos. Com as PPT, MTQ e ficheiro *CAD* das medições consegue identificar o local exato em que cada artigo foi considerado pelo medidor, podendo assim a) planear o seu custo em função do contexto no projecto e, com igual importância, b) compreender eventuais erros de medição. Ora, no caso de MTQ elaborados de forma automática a partir dos modelos de informação, não existe esse *CAD* de interface entre medidor e orçamentista. É por isso que é fundamental conferir-lhe ferramentas para executar essas tarefas a) e b) além das PPT e do MTQ de resumo de quantidades. Assim, um *dashboard* de apoio à orçamentação sem componente gráfica, i.e., sem acesso visual ao modelo tridimensional, não satisfaz o pretendido. Pode-se até dizer que pouco acrescenta às PPT. Com isto em mente, esta abordagem cai por terra e, à falta da capacidade nativa do *Power BI* de importação de modelos de informação, foram investigados softwares extra para possibilitar essa incorporação.

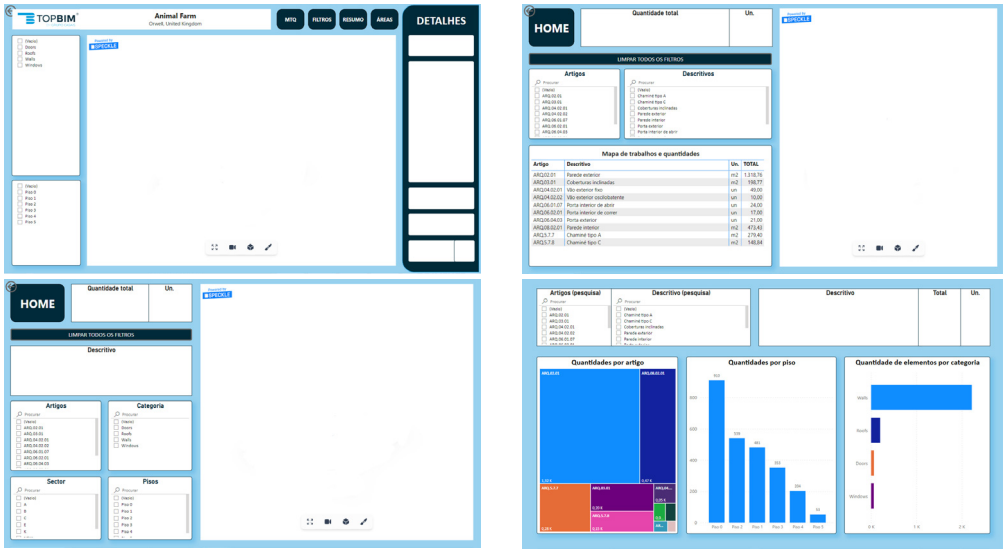

Figura 2
Dashboard "Técnico orçamentista" (da esq. para a dir., de cima para baixo: página geral, página MTQ, página filtros avançados, página resumo. Tracejado: espaço para 3D).

5. Ferramentas de integração de modelos de informação em Power BI

Da investigação elaborada, surgiram algumas ferramentas que estabelecem a ponte entre informação de projecto (geometria 3D + base de dados) produzida em *Revit* e o *Power BI*. O primeiro passo foi aferir se as funcionalidades descritas pelos fabricantes seriam compatíveis com as pretendidas e quais as eventuais limitações. Segue-se o resumo desta análise.

Da análise exposta (não exaustiva de toda a pesquisa efetuada), foi identificado o *Tracer* V3 (da *Proving Ground*) como o que potencialmente melhor responderia aos critérios pretendidos. A aferição das funcionalidades – e respetivos veredictos - passou por utilização das versões trial de cada programa. Foi, então, dada prossecução a essa implementação, conforme será descrito na próxima secção. Nota: à data da pesquisa, a opção *Speckle* passou despercebida. Só com o avançar do ano e em face da constante publicação em redes sociais sobre as mais valias da ferramenta (interoperabilidade entre softwares 3D), se materializou a sua exploração.

Tabela 1: Análise de softwares para elaboração de *dashboards* BIM

Software	Características	Licença	Veredicto
VCAD (VDT)	Vocacionado para *Health Check* dos modelos. Só para modelos em cloud (ex.: ACC).	Pago	Não testar
Power BIM	Apenas para modelos em *cloud*. Sem interação com modelo, apenas exposição.	Pago	Não testar
3DBI	Não permite *templatização*. Cada atualização do modelo exige recriação do *dashboard*.	Pago	Não testar
Tracer V3	Cumpre os pré-requisitos.	Pago	Testar
Speckle	Cumpre os pré-requisitos.	Open source	Testar (posterior)

6. Abordagem 2: Tracer + Power BI

O *Tracer* é um *plug-in* do *Autodesk Revit* que permite extrair a informação dos modelos de informação através de uma interface própria e que compila geometria e dados num ficheiro de base de dados (*.db*). Ao importar esse ficheiro no *Power BI* como ODBC *database*, cada elemento exportado contém toda a informação paramétrica e uma geometria (*mesh*) que o *Power BI* consegue processar e expor. O facto de se conseguir simplesmente alterar a origem de dados alterando o caminho para o ficheiro base ODBC – mantendo a configuração dos *dashboards* - e a geometria do modelo ser importável para um *visual* anexo ao programa, satisfaz os pré-requisitos. O *workflow* começa no *Autodesk Revit*, no qual surge na *taskbar* um menu *Proving Grounds*, com o *plug-in Tracer*. Surge então uma interface na qual existem várias opções de exportação, como o nível de detalhe (*coarse, medium, fine*) ou ainda o refinamento da geometria a exportar (de ressalvar a hipótese de exportar geometria demasiado complexas como *bounding boxes*, particularmente útil para otimizar tempo de exportação). Podem ser também selecionadas quais as categorias de elementos a exportar. No final, é possível gravar a configuração destas opções como um *preset* para facilitar futuras exportações. À cabeça, é identificável um potencial problema: apesar de as categorias poderem ser pré-selecionadas no *preset* ou "filtradas conforme a vista atual do modelo", não existe a opção de exportar apenas os elementos visíveis em determinada vista. Ou seja, a filtração dos elementos para determinada categoria a exportar é binária, ou seja, ou nenhum ou todos os elementos que existam no modelo, independentemente da vista. Este fator não favorece uma seleção mais criteriosa dos elementos e podem inclusivamente vir a constar do modelo geométrico importado para o *Power BI* artefactos geométricos difíceis de rastrear no modelo *Revit*.

O ficheiro exportado é um ficheiro *database* (db) que agrega toda a informação geométrica e não geométrica, que depois é simplesmente importado para o *Power BI* escolhendo como fonte de dados um ficheiro tipo ODBC, com a particularidade de ter de se escolher quais das tabelas de elementos se pretende importar. Para além desta segregação se ter revelado problemática, da abordagem com o *Tracer* emergiram outras dificuldades, algumas efetivamente incontornáveis e que levaram ao abandono da solução. Segue-se a descrição desses temas.

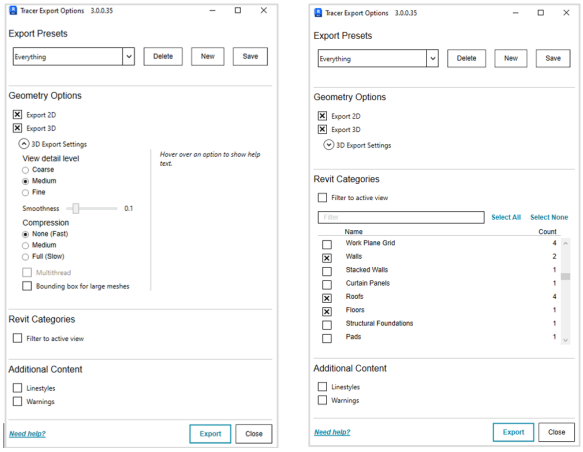

Figura 3
Tracer V3, opções de exportação a partir do *Autodesk Revit*.

6.1. Problema 1: exportação/importação segregada dos elementos

As tabelas são segregadas por categoria de elementos *Revit*, com a exceção de algumas gerais (não é claro o critério efetivo). Nas tabelas gerais, estão as geometrias e alguns parâmetros. Nas tabelas de cada categoria estão todos os parâmetros, incluindo os pretendidos do MTQ (artigo, descritivo, unidades, quantidades), o piso e o sector em que está o elemento, todos estes previamente inseridos no *Revit*. Assim, para podermos relacionar a tabela da geometria com as dos parâmetros, para que um parâmetro de uma afete a outra, é necessário fazer uma terceira, nova, de compilação de todas as categorias importadas e suprimir os parâmetros irrelevantes (através do *Power Query* - programa agregador das consultas externas que corre dentro do *Power BI* e permite algum tratamento dos dados). Os parâmetros desta tabela serão, então, os que se podem usar para filtrar os *visuals* (artigo, piso, sector, etc.). Ora, apesar de esta abordagem funcionar, é incompatível com a *templatização* do *dashboard* porque, na ausência de qualquer uma dessas tabelas de categorias no ficheiro de base de dados (se não existirem essas categorias nos modelos originais), o *Power Query* não processa as tabelas vazias, emitindo um erro na obtenção de dados. Teria de ser revisto a cada importação que elementos existem ou não no modelo e importadas ou não essas tabelas e recriada a tabela de compilação. Isto está longe de ser ideal. Como foi contornado: após muitos testes na tentativa da flexibilização da compilação dos dados através de *Power Query* independentemente das tabelas existentes na origem (base de dados), só se conseguiu resolver a possibilidade de *templatização*, ou seja, utilização do mesmo *dashboard* independentemente das variações dos ficheiros da origem de dados, através de uma fonte de dados extra produzida em *Revit*. Optou-se por elaborar um ficheiro *csv* com toda a informação que se tentara, sem sucesso, compilar diretamente no Power BI, nomeadamente com ID dos elementos, parâmetros de MTQ, de piso e de sector. Esse ficheiro seria então também importado para o *Power BI* e estabelecida a relação entre uma única tabela da base de dados *Tracer* que contivesse a *mesh* (geometria) e o ID dos elementos. Desta forma ficava fechada a relação geometria (*Tracer*) ⇔ ID ⇔ informação MTQ, Piso, Setor (csv). O problema original é que essa tabela do *Tracer* não continha os parâmetros pretendidos para todos os elementos. Por exemplo, para todos os parâmetros de uma parede, teria de ser consultada a tabela *Walls* e daí a necessidade original de compilação de todas as tabelas com informação parcial para criar uma única cujo parâmetro "Artigo" do MTQ se pudesse utilizar para filtrar os *visuals* independentemente da categoria de elemento.

6.2. Problema 2: várias designações Revit para "Piso"

Dependendo da categoria de Revit, cada elemento pode ter uma designação diferente para identificar o piso. Nas *Walls*, chama-se "*Base Constraint*", num *Floor* é "*Level*", num *Structural Column* é "*Base Level*". Ora, para usar o mesmo parâmetro "Piso" para filtrar o modelo, é necessária uma forma única de designação do piso, independentemente da categoria.

Como foi contornado: inicialmente, optou-se por acrescentar todas as colunas das várias categorias à tabela centralizadora referida no problema anterior, pela criação de uma nova coluna com o nome "Piso" e pela programação de uma fórmula que fosse, para cada elemento buscar o valor (ID do piso) a qualquer dessas várias colunas (*Base Constraint*, *Level*, etc.) que tivesse valor. Assim, se automatizou a atribuição de um parâmetro centralizador para o piso dos elementos, para utilizar nos filtros. Contudo, após a introdução do novo ficheiro de *Dynamo* (ver problema 1), optou-se por incluir já nesse esta informação, resolvendo o tema a montante.

Figura 4
Coluna "Piso" para resumo das várias designações conforme a categoria *Revit*.

1.2 Base Constraint	1.2 Base Level	1.2 Level	1.2 Reference Level	1.2 Piso
355	0	0	0	355
0	0	2607	0	2607
0	0	355	0	355
0	0	0	355	355

6.3. Problema 3: impossibilidade de exportação de modelos com links

O *Tracer* exporta apenas o modelo atual de Revit, isto é, não são exportados os modelos vinculados ou em *link*, a não ser como instâncias de uma tabela de ficheiros em *link* (*link* 1, *link* 2, etc.) que apenas identificam o nome do mesmo e pouco mais. No piloto testado, tratando-se de uma residência universitária com inúmeros quartos idênticos, optou-se pela modelação de cada tipologia como um *link*. Ou seja, a base de dados do modelo de arquitetura dependia obrigatoriamente da inclusão destes *links*, algo que não é possível com o *Tracer* atual. Como foi contornado: não é possível contornar este problema na atual versão do *Tracer*. No exemplo supra, implicou fazer *bind* aos modelos em *link* antes de exportar. Este passo é muito demorado e sujeito a erros e como tal impede a agilidade de atualização em função de alterações ao modelo (1 dia para fazer *bind*). Ainda assim, considerou-se que, apesar de não ser ótimo, este problema poderia não inviabilizar a metodologia através do *Tracer*, visto ter sido contornado.

6.4. Problema 4: ausência de contexto no modelo 3D filtrado

Aquando da aplicação de filtros ao modelo, p.ex., para só ver elementos com determinado valor de "Artigo", apesar da agradável experiência de aproximação automática a esses elementos, é subtraído o contexto. Este comportamento não é possível contornar, é característico do visual do *Tracer*. Se para os anteriores problemas foram encontradas soluções, este fator foi crítico para o abandono da solução. A inteligibilidade do local de aplicação foi a premissa base para a criação dos *dashboards* de apoio à orçamentação. Sem este, o *dashboard* não faria sentido.

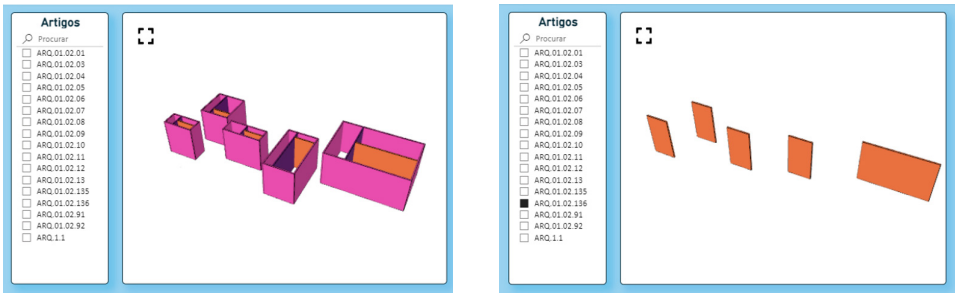

Figura 5
Modelo completo (esq.) / filtrado (dir.); ausência de contexto do visual do Tracer.

7. Abordagem 3: Speckle + Power BI

A utilização do *Speckle* permitiu finalizar uma primeira solução viável para implementação da metodologia. O *Speckle* atua como um conversor de toda a informação dos modelos de vários softwares (geométrica e não geométrica) para um modelo na linguagem *Speckle*, alojado no seu servidor. Pode-se, então, referenciar esse modelo como *links*, em páginas web ou, neste caso, via *Power BI*. Uma vez que a importação é simplesmente feita através da substituição do URL, pode-se substituir a origem de dados para utilizar num *template* previamente configurado.

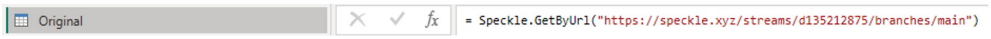

Figura 6
Determinação da origem de dados *Speckle* no *Power Query*.

Em relação à abordagem 2 com o *Tracer*, o *Speckle* apresenta algumas diferenças que resolvem muitos dos problemas do primeiro:

- É *open source* e tem, também por isso, uma ampla gama de utilizadores, casos de estudo, documentação, vídeo-tutoriais e uma comunidade de apoio notável;
- Possui diferentes opções de exportação em relação ao *Tracer*, podendo-se exportar, p. ex., só determinada(s) vista(s) ou categoria(s). Assim, é possível um controlo muito superior de que elementos se pretende exatamente exportar. Porém, não é possível, como no *Tracer*, controlar o nível geométrico da exportação para além do "*coarse/medium/fine*", algo que poderia melhorar a performance na receção dos modelos *Speckle* ou o tempo de exportação;
- Se a vista a exportar contiver *links*, é possível ativar a sua inclusão na emissão para o servidor, ou seja, não é necessário fazer *bind* (por oposição ao ponto 6.3);
- Após filtração dos elementos, o visual do *Speckle* navega e aproxima-se do que se pretende realçar, preservando, contudo, o contexto geral do modelo. Conforme referido no ponto 6.4, isto é fundamental para uma experiência de utilização que vai de encontro ao pretendido.

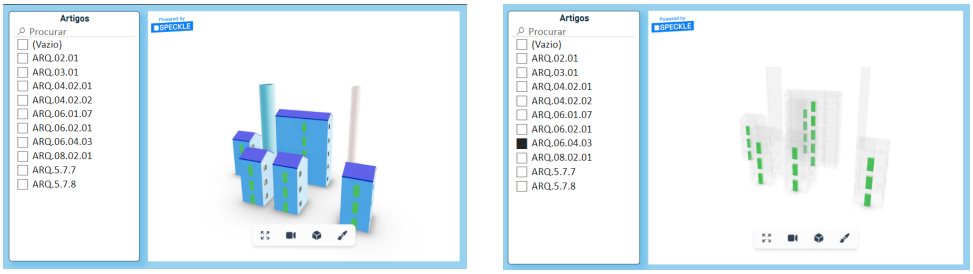

Figura 7
Modelo completo (esq.) / filtrado com contexto do *visual* do *Speckle* (dir.).

Apesar das melhorias apresentadas, surgiram ainda dois desafios que vale a pena mencionar.

7.1. Problema 1: acesso a parâmetros específicos dos elementos

A conversão dos parâmetros *Revit* na linguagem *Speckle*, cria uma hierarquia de parâmetros complexa, na qual consta toda a informação, mas que é de difícil acesso no *Power BI*. Apenas uma tabela é criada na consulta ao modelo, que contém todos os parâmetros. Alguns no primeiro nível (A) de "record" (ID, categoria, etc.) mas a maioria como *nested* "records" (B):

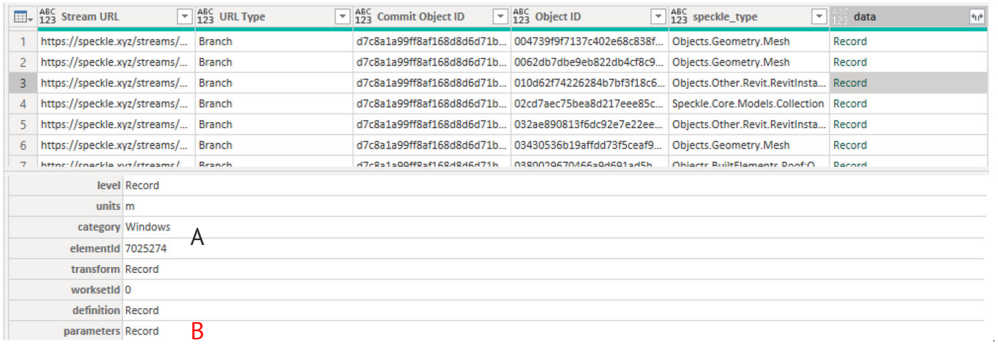

Figura 8
Consulta do *Speckle* no *Power BI*. Todos os dados são "*Records*" na coluna "*data*".

Como foi contornado: para alcançar os parâmetros de MTQ e Sector, foi necessário "explodir" os *records* da *data* até alcançar a informação relevante. Os nomes dos parâmetros são consultáveis no visualizador online do *Speckle*, uma vez que há lugar a uma codificação na qual não mantêm o nome original. Ainda assim, um fator crítico para não invalidar a *templatização*, é que aparentemente o mesmo nome de parâmetro em vários modelos diferentes, é convertido para o mesmo código de parâmetro *Speckle*, o que possibilita que se mantenham os passos do *Power Query* e não seja necessário reconfigurar todas as primeiras importações para *Power BI*.

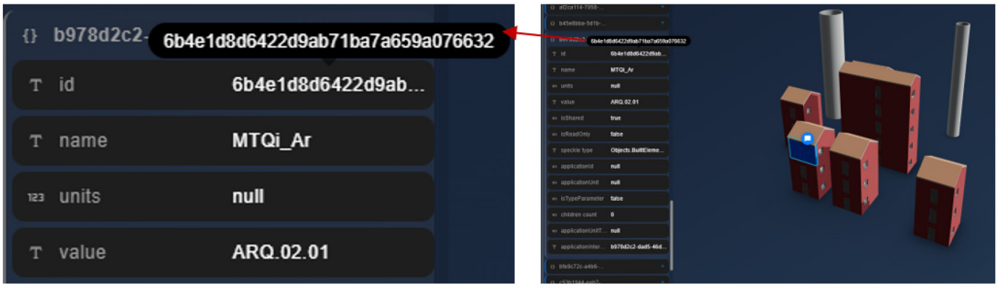

Figura 9
Parâmetros *Revit* vs. parâmetros *Speckle* (ex.: parâmetro "MTQi_Ar" = "Artigo").

Assim, acrescentando no *Power Query* a fórmula de expansão dos r*ecords* para os parâmetros que se sabe corresponderem aos pretendidos (MTQ), expandimos essa consulta num só passo (Figura 10). A partir deste momento, podemos utilizar os parâmetros expandidos para aplicar como filtros ao visual da geometria importada, como os de MTQ, sector e piso:

```
= Table.ExpandRecordColumn(#"Colunas Removidas1", "parameters", {"b978d2c2-dad5-46d3-936b-a0c1cb925ab4","bfe9c72c-a4b6-426f-be73-68ae52750cca",
    "a097e0dd-9c07-4440-bcfd-6f4d1855488b","b45e8bba-5d1b-41f1-8dfd-6b3de76d2035","90740b12-7c88-4c08-b57f-3ab32ca029ae","4df53ed0-7548-463e-a7cf-645e6357fb6f"},
    {"b978d2c2-dad5-46d3-936b-a0c1cb925ab4","bfe9c72c-a4b6-426f-be73-68ae52750cca","a097e0dd-9c07-4440-bcfd-6f4d1855488b","b45e8bba-5d1b-41f1-8dfd-6b3de76d2035",
    "90740b12-7c88-4c08-b57f-3ab32ca029ae","4df53ed0-7548-463e-a7cf-645e6357fb6f"})
```

Figura 10
Fórmula de expansão dos records dos parâmetros.

7.2. Problema 2: performance

Por fim, após a implementação bem-sucedida, resta mencionar um problema para já incontornável: performance do *visual* do modelo 3D. Se, para modelos pequenos, não é assinalável o tempo que demora a carregar o modelo dentro do Power BI a cada operação, para modelos grandes, é significativo. Para o modelo com cerca de 12000 elementos testado, o visual do modelo precisa de cerca de 1 minuto para o carregamento inicial e 2 minutos para aplicação de um filtro. Não retira a funcionalidade, mas é um inconveniente para a experiência de utilização. Este fator era idêntico (ou pior) na abordagem com o *Tracer V3*.

8. Outros processos resolvidos para o sucesso dos *dashboards*

De seguida, são descritas algumas dicas implementadas independentemente da abordagem que contribuíram para a otimização dos *dashboards* em Power BI.

- Não mostrar quantidades no visual de quantidades até haver algum filtro selecionado: criou-se uma *measure* associada à filtração ou não de determinados parâmetros. Depois alterou-se a cor do texto do cartão de resumo das quantidades para se o valor dessa *measure* fosse "1" (filtro ativo) ser preto; se fosse "0" (não haver filtros ativos) ser da cor do fundo (branco). Desta forma, se não houver nenhum filtro, o cartão de quantidades parece vazio.
- Inconformidades sistemáticas da origem: imaginando uma situação em que para um mesmo "Piso 1", haja elementos que o identifiquem como "Piso 1",

"piso 1" ou "PISO 1". Não havendo tempo para solicitar revisão ao modelo, podemos automaticamente procurar e substituir valores via *Power Query*, neste caso passando todos para "Piso 1", a constar dos filtros dos pisos, para não confundir o utilizador no momento da consulta.

9. Conclusão e futuros desenvolvimentos da solução encontrada

Conforme descrito, o piloto foi bem-sucedido para o perfil do *dashboard* de técnico de orçamentação, com exceção dos problemas de performance. Essa poderá eventualmente ser resolvida pela subdivisão dos modelos a constarem individualmente nos *dashboards* (4 páginas com 3000 elementos em vez de 1 com 12000). O *dashboard* é então partilhado com terceiros que possuam licença de *Power BI Pro* para seu usufruto em *browser*, quer via *smartphone*, *tablet* e/ou computador. Agora que está materializada a prova de conceito e o *workflow*, serão explorados outros *dashboards* para diferentes perfis de utilização, como:

- Promotor imobiliário: a partir de modelos com baixo nível de detalhe, apenas com *Rooms* classificados quanto ao programa funcional, colocar na mão do promotor um *dashboard* que apresente os *Rooms* em 3D com os KPI relevantes calculados automaticamente (total de cada tipologia funcional – quarto, sala, circulação, etc -, rácio áreas privadas vs. comuns, área de fachada, área de construção, etc.). Em complemento, poder colocar uns *sliders* à sua disposição para atribuir um preço/m^2 a cada tipologia e estimar um valor global para a obra;
- BIM manager/coordenador: *dashboards* de *Model Health*, para aferição do grau de "saúde" dos modelos de informação, com indicadores do verde ao vermelho, nomeadamente no que toca a parâmetros como número de avisos, contagem de famílias purgáveis ou *in-place*, vistas que não constem de folhas, etc. Ou ainda para verificação da qualidade da informação do modelo vs. *BIM Execution Plan*, ou seja, p.ex., análise das nomenclaturas dos ficheiros, desenhos, famílias, materiais, etc. Desta forma, poderiam ser mais facilmente auditados os modelos, simplesmente exportando periodicamente o ficheiro ".db";
- Direção de obra: semelhante ao orçamentista, mas mais orientados para consulta do modelo em tablet, encomenda de materiais por piso/sector, apoio aos autos, etc.

São estes os primeiros e os próximos passos da TOPBIM neste tópico com tanto potencial, sempre no sentido da democratização do BIM e da diminuição da informação BIMplícita.