Clientes Web OLAP baseados no MS Analysis Services: Uma Análise Comparativa entre Soluções Fat-Client e Thin-Client

Carlos Rui Gouveia Carvalhal

***Resumo* - Resumo: Este artigo inicia-se com uma análise comparativa, em termos de funcionalidades e limitações, entre as Soluções Fat-Client OLAP e Thin-Client OLAP. Seguidamente analisa uma série de Soluções Cliente OLAP, baseadas na Web, e suportadas pelo Analysis Services. Neste sentido, é analisada uma Solução Fat-Client OLAP baseada nas OWCs, no PivotTable Services e na Plataforma .NET, e três Soluções Thin-Client OLAP, uma baseada na ferramenta Analysis Services Thin Web Client Browser, disponibilizada no MS SQL Server 2000 Resource Kit, uma outra baseada na utilização das OWCs do lado do Servidor, e, finalmente, uma solução baseada no XMLA. Em relação a esta última Solução, é feita uma introdução e analise detalhada do Mecanismo de Acesso a Fontes de Dados Multidimensionais XMLA, onde são cobertos os seus Métodos e a sua implementação por parte do Analyis Services.**

***Abstract*: This paper begins with a comparative analysis, in terms of functionalities and limitations, between the Fat-Client and Thin-Client OLAP Solutions. Later, the paper analyzes some, Web-based, OLAP Client Solutions supported by Analysis Services: one Fat-Client and three Thin-Clients. The Fat-Client OLAP Solution analyzed is based on OWC, PivotTable Services and the .NET Platform. One of the Thin-Client OLAP Solution is based on a MS SQL Server 2000 Resource Kit tool, the Analysis Services Thin Web Client Browser. Other of the Thin-Client OLAP Solution is based on the use of the OWC, as an in-memory object, on the Server side. The last Thin-Client OLAP Solution analyzed is XMLA based. Related to this last Solution, the paper introduces and analyzes, in detail, the XMLA, as a Multidimensional Data Source Access Mechanism, covering its Methods and its implementation on Analysis Services.**

I. INTRODUCÃO

Sobre a arquitectura e funcionalidades do Analysis Services é possível implementar uma diversidade de Soluções Cliente OLAP, para funcionarem num ambiente Web. Soluções essas que tanto podem ser Thin-Client OLAP como Fat-Client OLAP, tendo cada uma destas categorias vantagens e desvantagens, e devendo a escolha entre uma e outra corresponder a uma solução de compromisso entre performance (em termos de tempo de resposta e funcionalidades analíticas) e acessibilidade (em termos de tipo/configuração da plataforma de acesso e infra-estrutura de rede física/lógica utilizada), conforme analisado mais adiante neste artigo.

É a diversidade de Mecanismos de Acesso disponibilizados pelo Analysis Services e a sua flexibilidade que possibilita um tão grande leque de possibilidades de implementação. De facto, uma solução Thin-Client OLAP pode ser implementada usando qualquer dos três Mecanismos de Acesso disponibilizados pelo Analysis Services: Acesso Directo, Acesso via HTTP/HTTPS e Acesso via XMLA. É claro que os dois primeiros mecanismos só produzem soluções Thin-Client OLAP se o PivotTable Service 8.0 e o MDAC 2.6 forem deslocalizados, da sua posição típica no Cliente, para o Servidor Web. Uma solução Fat-Client OLAP, ao requerer a presença do PivotTable Services 8.0 e do MDAC 2.6 do lado do Cliente, só pode ser implementada usando o Mecanismo de Acesso via HTTP/HTTPS, ou, eventualmente, usando o Mecanismo de Acesso via XMLA (se forem usadas, do lado do Cliente, as OWCs para interacção com o Cliente e se se desenvolve, no Cliente, um agente analítico "poderoso", com capacidade de cache por exemplo). A plataforma [4] é um caso de uma solução Fat-Client OLAP implementada com base no XMLA.

II. FAT-CLIENTS OLAP VERSUS THIN-CLIENTS OLAP

No contexto Web OLAP, a designação Thin-Client OLAP refere-se às soluções OLAP nas quais toda, ou quase toda, a aplicação reside no Servidor, inclusive todas as funcionalidades de Visualização e Exploração OLAP, as quais são controladas e fornecidas a partir do Servidor, ficando o Cliente reduzido à simples exibição dos ecrãs, previamente elaborados do lado do Servidor, e à recolha, e posterior encaminhamento, do input do utilizador. Na prática, isso quer dizer que o Cliente OLAP resume-se a uma Página HTML Standard, exibida por um Browser Web, através da qual são apresentados Relatórios OLAP e são disponibilizados elementos de interacção capazes de recolher as intenções de Navegação/Exploração OLAP dos utilizadores (as operações de Navegação/Exploração OLAP, mesmo as mais rudimentares, são executadas no Servidor). É óbvio que uma solução deste tipo consome muita largura de banda e recursos do Servidor, e tem tempos de resposta, expectáveis, longos. No entanto, têm algumas vantagens e são as mais adequadas em certas situações, conforme referido na Tabela 1.

Numa solução Fat-Client OLAP uma parte significativa da aplicação reside no Cliente, nomeadamente todas as componentes necessárias à completa implementação de uma interface de utilizador adequada à Visualização e Exploração OLAP, interface essa que terá de ser, obrigatoriamente, interactiva. Nestas soluções, a actividade exploratória do utilizador é traduzida, pela aplicação cliente, em quesitos multidimensionais que poderão ser resolvidos do lado do Cliente ou encaminhados para o Servidor. De facto, algumas das soluções Fat-Client OLAP dispõem de uma cache, onde são guardadas as respostas aos quesitos mais recentes (ou mais frequentes), o que, associado a uma entidade com "Inteligência OLAP", permite resolver localmente (i.e., sem recorrer ao Servidor OLAP) alguns dos quesitos do utilizador, funcionalidade que tem um impacto notável sobre o tempo de resposta. No entanto, estas soluções têm algumas desvantagens, não sendo, inclusive, as mais adequadas em certas situações, conforme referido na Tabela 1. Estas soluções são implementadas, sobre o Browser Web, recorrendo a JavaBeans, Applets Java, Office Web Components, Controlos COM/ActiveX e outros Plugins/Componentes.

III. SOLUÇÃO FAT-CLIENT OLAP BASEADA NAS OFFICE WEB COMPONENTS E NA PLATAFORMA .NET

As Office Web Components (OWC), a plataforma .NET e o MS Analysis Services possibilitam o desenvolvimento de Soluções Analíticas OLAP baseadas na Web e capazes de satisfazer plenamente os requisitos espectáveis para uma plataforma desta natureza, conforme descrito em [11].

Um exemplo de uma solução deste tipo é a descrita em

Tabela 1: Análise comparativa entre as soluções Fat-Client OLAP e Thin-Client OLAP.

ASPECTO	THIN-CLIENT OLAP	FAT-CLIENT OLAP
Tempo de Resposta	Apresenta um Tempo de Resposta elevado, pois toda e qualquer acção do utilizador sobre a interface requer um acesso ao Servidor Web e, eventualmente, também ao Analysis Server (por exemplo, se tivermos, do lado do Servidor, uma OWC PivotTable, a funcionar como objecto em memória, e um PivotTable Services, algum dos quesitos poderão ser resolvidos sem recorrer ao Analysis Server, desde que os dados necessários estejam disponíveis do lado do Servidor Web). Notar que, no entanto, o acesso ao Servidor Web (inclui também o tempo de resposta deste) terá, na maior parte dos casos, um peso bastante significativo no Tempo de Resposta.	O Tempo de Resposta é inferior ao de uma solução Thin-Client OLAP, obtendo o utilizador, frequentemente, respostas instantâneas, desde que os dados para responderem ao(s) quesito(s) estejam disponíveis do lado do Cliente (por exemplo, nos dados da própria OWC PivotTable, integrante da interface do o utilizador, ou na cache do PivotTable Services). Notar que, no entanto, o tempo de reposta inicial é elevado, pois é necessário carregar e instalar as componentes que implementarão a Aplicação Cliente/Interface de Utilizador (JavaBeans, Applets Java, Office Web Components, Controlos COM/ActiveX e outros Plugins/Componentes) e os seus dados iniciais.
Consumo de Recursos na Máquina Cliente/Servidor	Os Recursos consumidos do lado do Cliente não são significativos, o que já não se aplica ao Servidor, pois todo (ou quase todo) o processamento, inclusive todas as funcionalidades de Visualização/Exploração OLAP e transcrição da interacção do utilizador em quesitos multidimensionais, é controlado e realizado pelo Servidor. Não requer o carregamento nem a instalação de nenhuma componente adicional do lado do Cliente.	O processamento encontra-se distribuído entre o Cliente e o Servidor. O Cliente implementa sempre as funcionalidades de Visualização/Exploração OLAP e transcrição da interacção do utilizador em quesitos multidimensionais, podendo substituir, em certas situações, o próprio Analysis Server, fornecendo funcionalidades de criação/gestão de Cubos Locais e de armazenamento em cache dos quesitos mais recentes/frequentes. Requer a instalação, do lado do Cliente, de componentes especiais que implementarão a Aplicação Cliente/Interface de Utilizador.
Consumo de Largura de Banda	Há um grande volume de tráfego entre o Cliente e o Servidor, pois qualquer interacção do utilizador com a interface requer uma resposta por parte do Servidor, que sintetizará um novo ecrã e enviá-lo-á, como resposta, para o Cliente. Ecrãs esses que incluirão, na maior parte dos casos, um grande volume de imagens. (É possível ter uma solução que utilize, no Servidor, Componentes OWC PivotTable e Chart e mantenha no Cliente "imagens" destas componentes, usando para tal os seus métodos .ExportGif, que retornam numa imagem GIF a sua interface visível, mesmo quando são usados como objectos em memória sem interface gráfica. Para mais detalhes consultar [7].)	O carregamento das componentes que implementam a Aplicação Cliente/Interface de Utilizador gera um pico no tráfego com o Servidor (é carregado o código que implementa a aplicação/interface e os dados iniciais). No entanto, na fase de utilização uma solução Fat-Client OLAP gera menos tráfego que uma solução Thin-Client OLAP.
Compatibilidade Multi-Plataforma	Qualquer Browser Web a correr sobre qualquer Sistema Operativo serve de Cliente para uma solução Thin-Client OLAP, pois estas soluções utilizam somente o HTML Standard.	Nem todos os Sistemas Operativos ou Browsers Web são capazes de executar a Aplicação Cliente/Interface de Utilizador de uma solução Fat-Client OLAP, como é o caso daquelas que utilizam as OWCs, versão 11.0, que só funcionam no Sistema Operativo MS Windows 2000, ou posterior e no Browser Web IE 5.01, ou posterior, e requer que a máquina Cliente tenha instalado um produto do MS Office 2003. Para mais detalhes consultar [7].
Complexidade das Metodologias e Arquitecturas de Acesso	Quem acede ao Analysis Services é o Servidor Web, o que simplifica consideravelmente as Metodologias e Arquitecturas de Acesso, conforme é possível concluir de [8].	Dependendo da implementação, pode requerer Metodologias e Arquitecturas de Acesso, ao Analysis Services, complexas, as quais já foram oportunamente discutidas em [8].
Complexidade dos Mecanismos de Autenticação e Controlo de Acesso	Quem acede ao Analysis Services é o Servidor Web, o que simplifica consideravelmente os Mecanismos de Autenticação e Controlo de Acesso. Em [8] descrevem-se algumas situações nas quais é possível verificar isso.	Os Mecanismos de Autenticação e Controlo de Acesso são mais complicados, pois, na maior parte dos casos, quem está a aceder ao Analysis Services é o processo Cliente. Em [8] descrevem-se algumas situações nas quais é possível verificar isso.
Situações às quais melhor se Adequam	Adequam-se melhor a utilizadores ocasionais que não realizam processamento OLAP intensivo (por exemplo, só estão interessados em aceder a Relatórios OLAP pré-definidos sobre os quais muito raramente realização operações de Roll-Up, Drill-Down, Slice e Dice, e provavelmente nunca operações de Pivoting, nem alterarão os Membros das Dimensões que constam deles).	São os mais adequados às situações em que os utilizadores vão realizar processamento OLAP intensivo, nomeadamente alterar a estrutura de Relatórios OLAP pré-definidos, através de operações de Pivoting ou através da alteração dos Membros e Dimensões que nele constem.

[11]. Esse artigo apresenta uma solução OLAP, desenvolvida em ASP.NET, baseada no SQL Server 2000 Analysis Services, nas OWC e em XML Web Services, mostra como construir esta plataforma (o artigo desenvolve uma aplicação exemplificativa da solução proposta, sendo acompanhado do código necessário à sua implementação), descreve as configurações que é necessário aplicar ao Analysis Services (e IIS), para este poder ser acedido via HTTP/HTTPS (são as configurações descritas em [8], para uma Arquitectura com Configuração de Computador Único), e os requisitos que o IE tem de satisfazer, e as configurações que têm de lhe ser aplicadas, para conseguir aceder ao Analysis Services, via HTTP/HTTPS, recorrendo às OWC (algumas destas configurações já foram referidas em [8]). Ao longo desta secção analisar-se-á, de uma forma detalhada, esta arquitectura, e, superficialmente, a aplicação desenvolvida para exemplificar a sua implementação.

A.- Descrição da Arquitectura

A Figura 1 representa a arquitectura da solução descrita em [11]. Sobre esta arquitectura estão definidos 3 Trilhos para o fluxo de dados, que foram numerados pela ordem pela qual serão usados (na maior parte das vezes, pois pode-se iniciar uma sessão carregando um Relatório OLAP previamente gravado), pela aplicação, ao longo de uma Sessão de Análise. Os objectivos de cada um destes Trilhos são os seguintes:

- ***Trilho 1 (Ligação entre a OWC PivotTable e a Fonte de Dados OLAP):*** Permite estabelecer uma ligação entre uma OWC PivotTable (ou Chart, pois esta Componente também tem funcionalidades OLAP, conforme descrito em [7]) e o Analysis Services, via HTTP, através de XML Web Services.
- ***Trilho 2 (Quesitação Dinâmica e Interactiva da Fonte de Dados OLAP):*** Permite, à OWC PivotTable, inquerir, de forma dinâmica, interactiva e directa o Analysis Services (só intervém o PivotTable Service do Cliente e a Analysis Services HTTP Gateway, implementada sobre o Servidor Web). É por este Trilho que vão transitar os quesitos e dados associados ao Processamento Analítico Interactivo, razão pela qual é necessário, que a solução usada, garanta tempos de resposta reduzidos, daí a necessidade de diminuir o número de intermediários. Neste sentido, notar que, a solução que é proposta por Jeffrey Hassan e Kenneth Tu, em [11], não é a que é defendida aqui, mas sim o acesso representado a traço descontínuo na Figura 1, i.e., um acesso, de facto, directo entre o PivotTable Services Cliente e o Analysis Services, utilizando o Mecanismo de Acesso Directo, descrito em [8]. No entanto, devido às limitações do Mecanismo de Acesso Directo, descritas em [8], que impossibilitariam a utilização desta arquitectura nos acessos inter-domínio ou via Internet, propõe-se no seu lugar um acesso usando o Mecanismo de Acesso via HTTP, também descrito em [8]. É óbvio que a solução proposta tem tempos de resposta superiores, conforme descrito em [8], mas é uma solução de compromisso.
- ***Trilho 3 (Gravação/Recuperação de Relatórios OLAP):*** Permite gravar, numa BD Relacional, os Relatórios OLAP criados pelo utilizador, assim como também recuperar relatórios previamente gravados. Notar que aquilo que é guardado é um Documento XML (valor do atributo XMLData do Objecto PivotTable associado à Componente PivotTable) contendo os dados e o aspecto da Componente PivotTable, conforme descrito em [7]. Tais Documentos XML também poderiam ser guardado num ficheiro texto, no entanto, a solução da BD permite uma melhor gestão do acesso concorrente além de ser uma forma mais organizada de armazenamento. Notar ainda que os Relatórios OLAP também poderiam ser gravados do lado do Cliente, simplificando a arquitectura proposta, no entanto, tal solução seria limitativa em

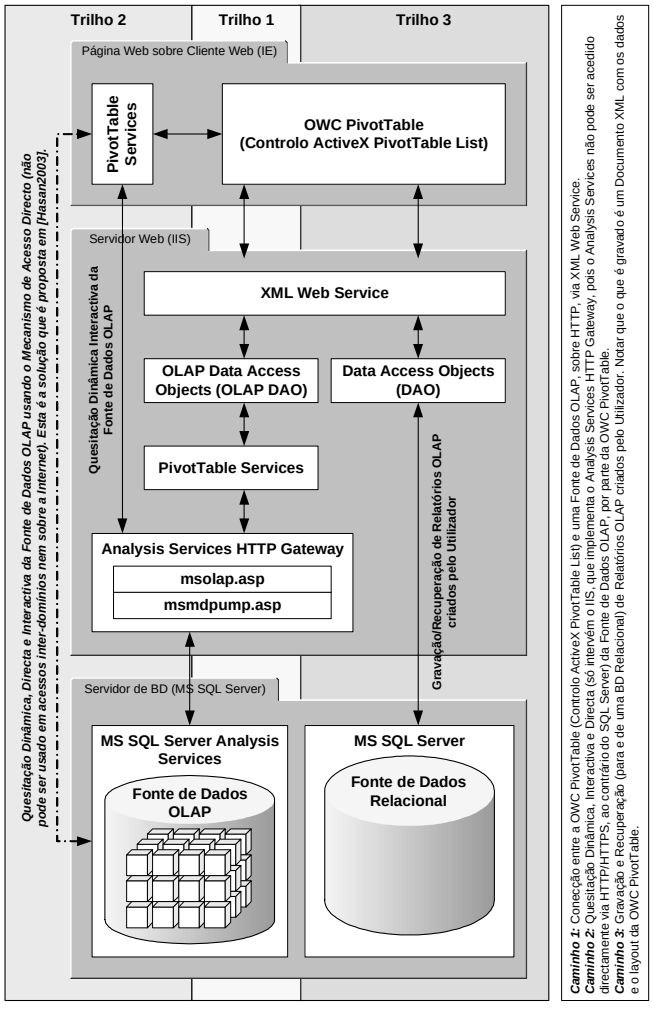

Figura 1: Arquitectura de uma solução OLAP, baseada na Web, implementada com recurso às OWC e à plataforma .NET. (Adaptado de [11]).

termos de mobilidade do utilizador e de partilha de relatórios entre utilizadores.

O XML desempenha um papel central nesta arquitectura, estando presente nos Trilhos 1 e 3. De facto, a possibilidade de carregar e formatar as OWC PivotTable através de dados XML, assim como também, a sua capacidade de exportar o seu conteúdo e aspecto no formato XML, possibilita o desenvolvimento, em ASP.NET, de XML Web Services que comunicam bidireccionalmente com o Controlo PivotTable List.

Assim sendo, no Trilho 1 a OWC PivotTable é carregada, e o seu aspecto definido, através de dados XML sintetizados pelos XML Web Services, a partir dos dados extraídos da Fonte de Dados OLAP. No Trilho 2 os dados e o aspecto da Componente PivotTable são importado/exportados, em formato XML, de/para uma Fonte de Dados Relacional, também através de XML Web Services.

Outra das potencialidades das OWC, que é explorada nesta arquitectura, é a sua utilização do lado do Servidor (instanciadas como objectos residentes em memória, sem interface gráfica mas que disponibilizam todos os seus serviços, através do seu Modelo de Objecto). De facto, o bloco da arquitectura designado OLAP Data Access Objects cria, do lado do Servidor Web, uma instância de OWC PivotTable, fá-la estabelecer uma ligação com o Analysis Services e obter desta os dados que se pretende apresentar ao utilizador, configura a vista dos dados inicial, que se pretende apresentar, e usa esta instância para gerar os dados XML que vão ser carregados na OWC PivotTable do Cliente. É necessária a presença do PivotTable Services no Servidor Web para fornecer, à instância OWC PivotTable ali residente, acesso ao Analysis Services.

Uma vez estabelecida a ligação entre a Componente PivotTable do Cliente e o Analysis Services (notar que a configuração de conexão ao Analysis Services também está contida nos dados XML que foram exportados da instância Servidor para a instância Cliente da Componente PivotTable) o utilizador pode assemblar livremente, usando a interface de utilizador da Componente PivotTable, a vista de dados que melhor se adequa às suas necessidades. Sempre que as alterações efectuadas pelo utilizador sobre a vista de dados não puder ser resolvida pela Componente PivotTable, será, dinamicamente, criado e enviado para o Analysis Services, um query MDX que permitirá produzir a vista desejada. Notar que pode ser que o query MDX possa ser resolvido no PivotTable Services do Cliente, poupando, assim, o acesso ao Analysis Services (funcionalidade potenciada pela cache do PivotTable Services) [7]. Se tal não for possível, a query será encaminhada para o Analysis Services, acedido através do bloco Analysis Services HTTP Gateway. Esta é a lógica programática implementada através do Trilho 2.

B.- Breve Análise da Aplicação

A interface de utilizador da aplicação proposta em [11] é implementada através de um formulário Web, desenvolvido em ASP.NET (owc10.aspx), que contém uma OWC PivotTable (lado do Cliente) e um conjunto de botões para estabelecer a conexão com a fonte de dados OLAP e para gravar e carregar Relatórios OLAP do utilizador. Estas funcionalidades são fornecidas por um conjunto de Web Methods (XML Web Services) (wsOLAP.asmx) e acedidas, pela interface de utilizador, através de um conjunto de funções JavaScript (lado do Cliente) (olap.js). Estas funções JavaScript delegam a maior parte dos pedidos nos XML Web Services, fazendo chamadas, do lado do Cliente, aos Web Methos. A aplicação é acompanhada por um ficheiro DHTML Behavior (webservice.htc), que determina o comportamento e as funcionalidades dos controlos do formulário Web, que constitui a interface de utilizador. Para mais detalhe sobre a aplicação consultar [11].

IV. SOLUÇÃO THIN-CLIENT OLAP BASEADA NO ANALYSIS SERVICES THIN WEB CLIENT BROWSER

O Analysis Services Thin Web Client Browser é uma ferramenta, que vem incluída no MS SQL Server 2000 Resource Kit [2], que utiliza ASPs para aceder e explorar os Cubos de uma instância do Analysis Server, convertendo os dados multidimensionais em páginas HTML (elementos standards), as quais são passadas para o browser do Cliente. Esta ferramenta, conforme o seu nome indica, é da categoria Thin-Client OLAP e pode ser usada como base para o desenvolvimento de aplicações mais complexas, pois, apresenta-se sob a forma de um conjunto de Páginas ASP, que podem ser livremente editadas (é código livre, que vem incluído no MS SQL Server 2000 Resource Kit), ou ser usada tal como fornecida (com as necessárias configurações de acesso à instância pretendida do Analysis Server) na exploração dos Cubos OLAP do Analysis Services. A Figura 2 apresenta a arquitectura de uma solução baseada no Analysis Services Thin Web Client Browser.

O Analysis Services Thin Web Client Browser requer, do lado do Cliente, o IE 5.0, ou uma sua versão mais recente (este é um requisito que consta da documentação, no entanto, em princípio, pode ser usado com outros browsers Web compatíveis com as funcionalidades do IE 5.0). Porque o Cliente não estabelece uma ligação directa com o Analysis Services, não necessita do PivotTable Services

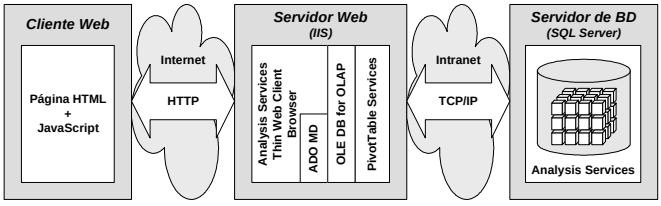

Figura 2: Arquitectura de uma solução baseada no Analysis Services Thin Web Client Browser.

8.0 nem do MDAC 2.6. O acesso ao Analysis Server é garantido através de Páginas ASP residentes no Servidor Web IIS. Como só esta instância do IIS é que tem de ter acesso ao Analysis Server (esse é um aspecto que é importante ter presente, pois vai ter influência sobre os mecanismos de autenticação de utilizadores e controlo de acessos que vão ter de ser definidos) é possível estabelecer uma ligação segura entre ambos, mesmo através de um firewall.

A interacção do Analysis Services Thin Web Client Browser com o Cliente é feita através de páginas HTML e código JavaScript, o que impõe algumas limitações em termos da "riqueza" da interface a conceber (comparativamente com aquilo que se consegue usando OWCs [7]). No entanto, a utilização do JavaScript vem minorar estas limitações, enriquecendo as páginas HTML, ao dotá-las do dinamismo necessário à implementação de uma interface de utilizador adequada aos Operadores OLAP.

Adicionalmente à dificuldade de implementação de uma interface de utilizador "rica" e funcional, o Analysis Services Thin Web Client Browser tem outras limitações, próprias de uma solução Thin-Client OLAP. Entre outras, pode-se referir o tempo de resposta, o qual é superior ao de uma solução baseada no PivotTable Services e nas OWCs.

Uma solução na qual é usado o Analysis Services Thin Web Client Browser é aquela que é descrita em [12]. Neste artigo, onde o autor faz uma análise de soluções baseadas nos Mecanismos de Acesso Directo e via HTTP, esta solução é apresentada como a mais adequada para a situação que ali se pretende resolver: acessos através de firewalls e sem necessidade de intervenção sobre as máquinas Cliente.

Se bem que sem usar o Analysis Services Thin Web Client Browser, os documentos [14, 15] descrevem como aceder ao Analysis Services usando a linguagem MDX e o ADO MD, através de ASPs, que é aquilo que acontece com o Analysis Services Thin Web Client Browser. Estes documentos vêm acompanhados de bastante código exemplificativo.

V. SOLUÇÃO THIN-CLIENT OLAP BASEADA NAS OFFICE WEB COMPONENTS

As OWC, nomeadamente a OWC PivotTable e a OWC Chart podem ser usadas do lado do servidor, instanciadas como objectos em memória, sem interface gráfica, controlados, programaticamente, através do seu modelo de objecto. Estas instâncias OWC disponibilizam, através do seu Modelo de Objecto, propriedades, métodos e eventos que permitem aceder aos seus "serviços" sem necessidade de interagir com a sua interface gráfica. Também disponibilizam propriedades/métodos que produzem uma representação estática da sua interface gráfica (que não está visível). No caso dos Controlos Chart e PivotTable List isto é feito pelo método .ExportGif, o qual retorna uma imagem GIF representativa da sua interface gráfica. [7]

No entanto, a utilização das OWC como objectos em memória é desencorajada pela Microsoft, pois há certas circunstâncias que podem provocar o bloqueio, instabilidade ou perda de desempenho da aplicação [1 (Chapter 1)]. No entanto, existem técnicas que permitem usar, com alguma segurança, as OWC como objectos em memória, conforme descrito em [1 (Chapter 13)].

Com base nessa funcionalidade das OWC Chart e PivotTable é possível implementar soluções Thin-Client OLAP, baseadas na arquitectura representada na Figura 3. Conforme pode-se verificar, esta arquitectura é semelhante à da solução baseada no Analyis Services Thin Web Client Browser, diferindo desta somente no facto de fazer uso das OWCs do lado do Servidor Web. No entanto, esta diferença tem efeitos significativos na qualidade da interface analítica que é apresentada ao Cliente, nas potencialidades analíticas da solução e na simplicidade de implementação das funções analíticas. De facto, o que é apresentado ao Cliente são imagens das interfaces das OWCs, bastante "ricas" e com funcionalidades analíticas poderosas. No entanto, são só imagens, pelo que terá de haver código (embutido na Página HTML) que recolha a interacção do utilizador e a reencaminhe para a instância OWC localizada no Servidor Web, a qual agirá em conformidade, alterando a sua interface visível, a qual terá de ser novamente enviada para o Cliente, sob a forma de uma imagem. Muito do processamento analítico, que na solução anterior era realizado exclusivamente pelo Analysis Services Thin Web Client Browser, passa a ser realizado pelas OWCs, o que simplifica a Aplicação, ao mesmo tempo que enriquece as funcionalidades analíticas da solução (para todas aquelas disponibilizadas pelas OWCs).

É óbvio que esta solução gera um grande volume de tráfego, tem longos tempos de espera e sobrecarrega o Servidor Web. No entanto, conforme já foi referido, tem vantagens significativas sobre a solução anterior, em termos daquilo que é disponibilizado ao Cliente. Uma solução deste tipo é aquela que é descrita em [13].

VI. SOLUÇÕES AD-HOC BASEADAS NO XMLA

A necessidade de instalação do PivotTable Services (que requer o MDAC 2.6, implicando na instalação de aproximadamente 14MB de software na máquina cliente, sem contar com a aplicação analítica, nem as eventuais OWCs), do lado do Cliente, para conseguir aceder ao Analysis Services, é um ponto fraco dos Mecanismos de Acesso Directo e via HTTP/HTTPS, limitando,

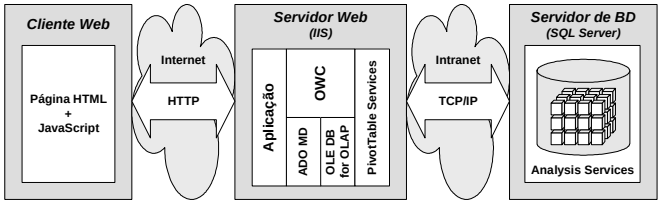

Figura 3: Arquitectura de uma solução Thin-Client OLAP baseada no uso das OWCs como objectos em memória do lado do Servidor Web.

nomeadamente, a plataforma que os utilizadores podem utilizar (Hardware e Sistema Operativo), o modelo de interface, a linguagem de programação e obrigando à manutenção da correspondência/compatibilidade entre as versões das componentes cliente (PivotTable Service e MDAC) e servidor (Analysis Server). A tentativa de ultrapassar este constrangimento através de soluções Thin-Client OLAP, nas quais o PivotTable Services é "deslocado" para o lado do Servidor Web, além de obrigar à utilização do IIS, como Servidor Web (com uma configuração bastante elaborada), produz soluções pouco performantes. Para ultrapassar estas limitações e responder à necessidade de um acesso universal às fontes de dados analíticas, OLAP e Data Mining, surgiu o XMLA, de XML for Analysis. (Notar que o XMLA é mais genérico do que isso, permitindo também o acesso às fontes de dados relacionais, no entanto, esta capacidade não vai ser explorada aqui.)

A.- Introdução ao XMLA

A especificação XMLA [6], proposta e suportada pela Microsoft e outras empresas líderes do mercado OLAP/Data Mining (agrupadas no XMLA Council), define uma interface Web Services standard (o XMLA é um standard industrial aberto), de acesso às fontes analíticas, OLAP e Data Mining, baseada na linguagem XML e nos protocolos Internet standards SOAP (Simple Object Access Protocol) e HTTP. Dessa forma, o XMLA permite às aplicações cliente analíticas, OLAP e Data Mining, inquerirem, de uma forma standard, sem recorrer a componentes específicas (como por exemplo, o PivotTable Services, essencial para o acesso ao Analysis Services nos Acessos Directo e via HTTP), as fontes de dados OLAP e Data Mining, e, aos fornecedores de dados analíticos, disponibilizarem interfaces de acesso universais, os XMLA Providers. Noutras palavras, através do XMLA, qualquer aplicação cliente OLAP/Data Mining passa a poder inquerir e explorar, de uma forma standard, qualquer fonte de dados OLAP/Data Mining, tal como acontecia com as BD Relacionais, baseadas no SQL. A Figura 4 apresenta a arquitectura genérica de uma plataforma XMLA, onde é possível verificar a distribuição das suas componentes entre as máquinas Cliente, Servidor Web e Fonte de Dados Analíticos. Chama-se a atenção, nesta arquitectura, para a ausência de qualquer componente do lado do Cliente (para além da aplicação analítica), e para a existência da componente XMLA Provider do lado do Servidor Web.

O XMLA é uma interface linguística, baseada em mensagens XML, que pode ser usada em qualquer plataforma, com qualquer linguagem de programação e com qualquer modelo de objecto, basta que suportem o XML e sejam capazes de executar métodos SOAP. Esta interface linguística, que é implementada como um Web Services, define uma linguagem standard, a mdXML, de quesitação de fontes de dados analíticas. Actualmente (especificação XMLA 1.1), a mdXML apresenta-se como uma versão, encapsulada em XML (elemento <Statement>) da linguagem MDX (de Multidimensional Expression Language, a linguagem de quesitação e manuseamento de BD Multidimensionais, e de definição de expressões multidimensionais, usada pelo Analysis Services –a interacção entre o PivotTable Services e o Analysis Server baseai-se na troca de mensagens MDX). No entanto, estão previstas extensões futuras, as quais serão sempre baseadas na MDX, a qual continuará a poder ser usada no XMLA através do elemento <Statement>. O papel desempenhado pela mdXML nos SGBD Multidimensionais é similar ao papel da SQL nos SGBD Relacionais. Tal como a estrutura mais comum no SQL é a instrução SELECT, a estrutura mais comum no mdXML também é a instrução SELECT. Mas ao contrário do Modelo Relacional, onde o resultado de uma instrução SELECT é uma Tabela (também designada Row Set), o resultado de uma instrução mdXML SELECT é um Cubo (também designado Cell Set).

A Tabela 2 faz uma análise comparativa entre as Arquitecturas de acesso ao Analysis Services XMLA, de Acesso Directo e via HTTP/HTTPS. Esta tabela aponta algumas desvantagens, relacionadas com o desempenho, que não devem ser vistas como uma proibição à sua utilização, pois estas limitações podem ser geridas durante a fase de desenho da aplicação. A título de exemplo, para reduzir o volume de tráfego com o servidor, a aplicação cliente pode manter em cache alguns dos metadados do

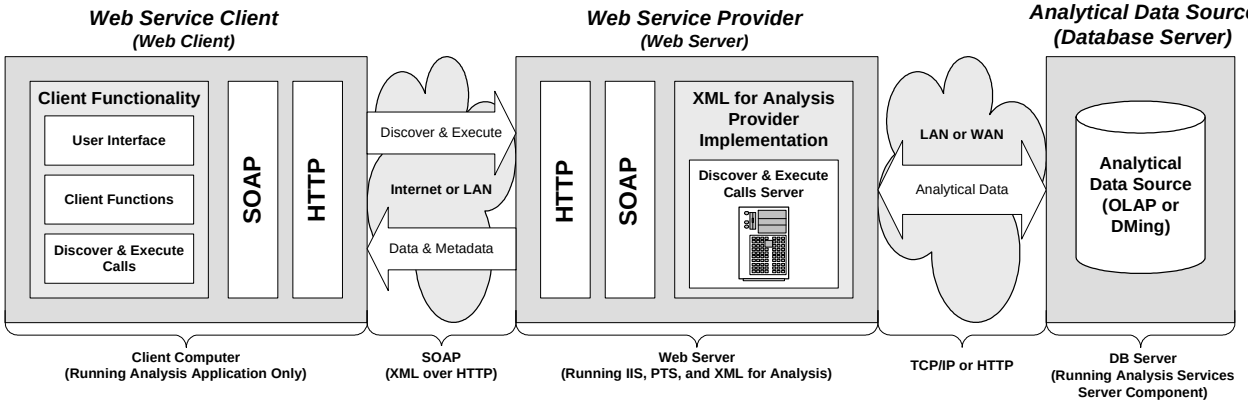

Figura 4: Arquitectura genérica de uma plataforma baseada no XMLA.

Cubo. Adicionalmente, como as aplicações analíticas das organizações têm uma grande variedade de perfis de utilizadores, pode-se adaptar a arquitectura da solução disponibilizada ao perfil dos utilizadores. É usual o maior grupo de utilizadores dos sistemas analíticos serem os menos exigentes, em termos de funcionalidades, podendo as suas necessidades serem satisfeitas através de relatórios estáticos ou parametrizados. Ao, frequentemente reduzido, grupo de utilizadores sofisticados, que requer, da aplicação, um alto desempenho e capacidades analíticas poderosas, a aplicação pode disponibilizar uma Arquitectura de Acesso Directo, limitada a ambientes Intranet ou VPN. Finalmente, ao grupo de utilizadores intermédio, que requer algumas capacidades exploratórias, mas, a liberdade dos acessos via Internet, pode-se disponibilizar acessos via XMLA ou HTTP/HTTPS.

A.1.- Métodos XMLA

Conforme já foi referido, o XMLA baseia-se no SOAP, um protocolo leve, baseado no XML, destinado à troca de Informação Estruturada através do protocolo HTTP. A Informação Estruturada além do seu conteúdo contém uma indicação do significado deste conteúdo. No caso do SOAP, isso é conseguido recorrendo a Tags XML. Dessa forma, uma Mensagem SOAP é um Documento XML que consiste num *SOAP Envelope* (o elemento XML raiz, que fornece um contentor para a mensagem), num *SOAP Header*, contendo informação específica da aplicação (por exemplo, informação de autenticação do utilizador), e num *SOAP Body*, que contém a mensagem que está a ser enviada. (Estas componentes da Mensagem SOAP podem ser verificadas nas Mensagens XMLA da Tabela 3 e Tabela 4.) Chamar um método SOAP não é mais do que embrulhar os seus argumentos em XML e enviar o pedido para o servidor, usando o protocolo HTTP. É isso que acontece quando é executado um Método XMLA: a instrução XMLA é encapsulada numa Mensagem SOAP que é enviada, usando os protocolos SOAP e HTTP, para o Servidor Web (Web Service Provider), onde está alojado o XMLA Provider, conforme representado na Figura 4, o qual instanciará o XMLA Provider. No XMLA Provider os Métodos XMLA são desencapsulados e executados, contra o fornecedor de dados analíticos. O Cliente recebe a resposta ao seu pedido sob a forma de uma Mensagem XMLA, encapsulada numa Mensagem SOAP, e via SOAP e HTTP, conforme representado na Figura 4.

Os XMLA Providers disponibilizam, somente, dois métodos:

- ***Discover:*** Este método XMLA é usado para obter metadados que descrevam os serviços suportados pelo XMLA Provider. É um método flexível que o Cliente pode usar repetidamente para "construir uma imagem" da configuração e capacidades do fornecedor de dados. Por exemplo, um cliente pode requerer primeiro a lista das fontes de dados que estão disponíveis a partir de um determinado servidor, e seguidamente inquerir sobre as propriedades e esquemas suportados por estas fontes, de modo ao cliente poder escrever e executar os quesitos adequados a cada uma delas. A Tabela 3 apresenta um exemplo de uma chamada ao Método Discover, apresentado as Mensagens Pedido e Resposta. Para detalhes sobre o formato e parâmetros destas Mensagens consultar [6, 9].

Tabela 2: Vantagens e desvantagens de uma Arquitectura XMLA relativamente às Arquitecturas de Acesso Directo e via HTTP/HTTPS.

VANTAGENS DO XMLA	DESVANTAGENS DO XMLA
A interacção entre o Cliente e o Servidor baseia-se num Protocolo Stateless: Isso significa que o Servidor Web não tem de lembrar-se dos clientes entre pedidos e, consequentemente, não tem de manter uma ligação à BD por cada cliente. Em vez disso, o Servidor Web pode manter uma pilha de ligações à BD que reutiliza na satisfação dos pedidos dos clientes. A manutenção de poucas sessões activas também significam uma menor sobrecarga no atendimento de um grande número de clientes.	***Baixa performance:*** Nos mecanismos de acesso ao Analysis Services Directo e via HTTP, o PivotTable Services mantém em cache os Metadados do Cubo (e também os resultados dos quesitos mais recentes ou frequentes), o que lhe permite responder rapidamente a numerosos pedidos da aplicação cliente. Numa configuração baseada no XMLA, cada um destes pedidos requer um acesso ao servidor.
Não é necessário instalar nenhum software especial na Máquina Cliente: Não é necessário instalar nenhum driver de ligação a BDs, como por exemplo o ADO-MD ou o OLE DB for OLAP (cuja instalação requer a instalação dos seus equivalentes relacionais ADO e OLE DB). A aplicação cliente só tem de ser capaz de usar o protocolo SOAP para comunicar com o XMLA. Uma aplicação tão simples como uma Página HTML, com scripts embebidas, pode usar o SOAP para comunicar com XMLA.	***As mensagens trocadas entre o Cliente e o Servidor são muito grandes:*** Estas mensagens são, provavelmente, dez vezes maiores do que as mensagens binárias equivalentes usadas pelo OLE DB for OLAP. Esta dimensão das mensagens é devida, por um lado, ao facto da informação ser empacotada como uma string (similar ao HTML), e, por outro lado, por cada componente da informação estar delimitada por Tags XML de abertura e fecho.
É independente da plataforma, linguagem de programação, e modelo programático e fornecedor de Web Services: É possível desenvolver aplicações analíticas cliente, usando o XMLA, em qualquer plataforma, usando qualquer linguagem de programação e modelo programático, desde que estes possibilitem a execução de métodos SOAP e suportem o XMLA. Esta independência também se estende ao Servidor Web, bastando para tal a existência de XMLA Providers para a fonte de dados analítica que se pretende utilizar. Isto permite, por exemplo, que o Analysis Services possa ser acedido a partir de outro Servidor Web que não o IIS.	

- ***Execute:*** Este método XMLA é usado para executar quesitos sobre uma fonte de dados analíticos, recebendo os dados retornados por este. Estes quesitos podem ser espremidos em MDX ou numa outra linguagem específica da fonte de dados. O quesito/comando a ser executado vai encapsulado no elemento XML <Statement>, conforme já foi referido atrás. A Tabela 4 apresenta um exemplo de uma chamada ao Método Execute, apresentado as Mensagens Pedido e Resposta. Notar que o que é transmitido na Mensagem Pedido é uma instrução MDX SELECT. Para detalhes sobre o formato e parâmetros destas Mensagens consultar [6, 9].

B.- Acesso ao Analysis Services via XMLA

O Analysis Services 2000 não suporta, de forma nativa, acessos via XMLA. Sendo esta funcionalidade assegurada através de uma Internet Server API (ISAPI), que tem de ser instalada num Servidor Web IIS que possa aceder ao Analysis Services via OLE DB for OLAP (i.e., tem de ter o PTS 8.0 e o MDAC 2.6 instalado) [17]. Esta ISAPI é fornecida pela Microsoft sob a forma de um Software Development Kit (SDK), designado XML for Analysis Software Development Kit (MS XMLA SDK), disponível em [5]. Este pacote de software, além do MS XML for Analysis Provider, contém exemplos de aplicações cliente e documentação. Os procedimentos necessários à instalação e configuração do MS XMLA Provider podem ser consultados na documentação que acompanha o SDK e nos documentos [9, 10].

O Analysis Services 2005 suporta o XMLA de forma nativa, pelo que não é necessário instalar o MS XMLA SDK [17]. Noutras palavras, o Analysis Services 2005 não necessita de XMLA Provider. Com o Analysis Services

Tabela 3: Exemplo de uma chamada ao Método XMLA Discover. Neste caso, o objectivo é obter a lista de todos os Cubos da BD FoodMart 2000. (Adaptado de [6]).

PEDIDO DISCOVER

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <Discover xmlns="urn:schemas-microsoft-com:xml-analysis"
   SOAP-ENV:encodingStyle=
           "http://schemas.xmlsoap.org/soap/encoding/">
   <RequestType>MDSCHEMA_CUBES</RequestType>
   <Restrictions>
    <RestrictionList>
     <CATALOG_NAME>
      FoodMart 2000
     </CATALOG_NAME>
    </RestrictionList>
   </Restrictions>
   <Properties>
    <PropertyList>
     <DataSourceInfo>
      Provider=MSOLAP;Data Source=local;
     </DataSourceInfo>
     <Catalog>
      Foodmart 2000
     </Catalog>
     <Format>
      Tabular
     </Format>
    </PropertyList>
   </Properties>
  </Discover>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

RESPOSTA DISCOVER ABREVIADA

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <DiscoverResponse xmlns=
                    "urn:schemas-microsoft-com:xml-analysis">
   <return>
    <root xmlns="urn:schemas-microsoft-com:xml-analysis:rowset">
     <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <!-- The XML schema definition of the result comes
           here -->
      ...
     </xsd:schema>
     <row>
      <CATALOG_NAME>FoodMart 2000</CATALOG_NAME>
      <CUBE_NAME>Sales</CUBE_NAME>
      ...
     </row>
     <row>
      <CATALOG_NAME>FoodMart 2000</CATALOG_NAME>
      <CUBE_NAME>Warehouse</CUBE_NAME>
      ...
     </row>
     ...
    </root>
   </return>
  </DiscoverResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Tabela 4: Exemplo de uma chamada ao Método XMLA Execute. Esta chamada a este método executa uma instrução MDX SELECT, a qual é transmitida através do comando <Statement>. (Adaptado de [6]).

PEDIDO EXECUTE

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <Execute xmlns="urn:schemas-microsoft-com:xml-analysis"
   SOAP-ENV:encodingStyle=
           "http://schemas.xmlsoap.org/soap/encoding/">
   <Command>
    <Statement>
     select [Measures].members on Columns from Sales
    </Statement>
   </Command>
   <Properties>
    <PropertyList>
     <DataSourceInfo>
      Provider=Essbase;Data Source=local;
     </DataSourceInfo>
     <Catalog>Foodmart 2000</Catalog>
     <Format>Multidimensional</Format>
     <AxisFormat>ClusterFormat</AxisFormat>
    </PropertyList>
   </Properties>
  </Execute>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

RESPOSTA EXECUTE ABREVIADA

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <m:ExecuteResponse
   xmlns:m="urn:schemas-microsoft-com:xml-analysis">
  <m:return
   SOAP-ENV:encodingStyle=
            "http://schemas.xmlsoap.org/soap/encoding/">
   <root xmlns=
         "urn:schemas-microsoft-com:xml-analysis:mddataset">
    <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:xars="urn:schemas-microsoft-com:xars">
      <!—The schema for the data goes here. -- >
      ...
    </xsd:schema>
      <!—The data in MDDataSet format goes here. -- >
      ...
   </root>
  </m:return>
  </m:ExecuteResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

2005, a aposta da Microsoft no XMLA vai muito mais longe do que isso, sendo o XMLA o protocolo nativo de comunicação, usado também nas comunicações via TCP/IP (lembrar que no Analysis Services 2000 as comunicações via TCP/IP utiliza o Protocolo SQL Analysis Services implementado pelo PivotTable Services). No entanto, o XMLA sobre TCP/IP é proprietário da Microsoft e não um standard aberto como o XMLA via HTTP. O documento [16] analisa esta funcionalidade em detalhe.

C.- Implementação de Clientes OLAP baseados no XMLA

Conforme já foi referido, o XMLA permite implementar Clientes OLAP capazes de acederem a qualquer fonte de dados analítica sem necessidade de instalar nenhuma componente específica do fornecedor de dados do lado do Cliente. O acesso às fontes de dados passa a processar-se de forma standard recorrendo aos Métodos XMLA e ao uso de uma linguagem de quesitação multidimensional, que, actualmente, tem de ser específica do fornecedor de dados, pois, a especificação XMLA ainda não define uma linguagem de quesitação de facto, delegando a execução dos quesitos no elemento <Statement>, que permite executar quesitos e transmitir instruções, para o fornecedor de dados, usando uma linguagem específica deste. Apesar deste facto, que não permite a quesitação universal, de facto, das fontes de dados analíticas, o Método XMLA Discover é standard, permitindo obter metadados e outras informações de qualquer fonte de dados multidimensional, que implemente o XMLA, e a linguagem MDX está integrada numa especificação aberta, podendo assim ser implementada por qualquer fabricante, o que está, de facto, a acontecer. Adicionalmente, é previsível que nos próximos tempos surja uma nova versão do XMLA, que inclua uma especificação mais completa da mdXML.

Como estes Clientes OLAP XMLA poderão ser implementados? Isto dependerá da complexidade da aplicação. Uma simples página HTML contendo scripts pode implementar um Cliente OLAP XMLA, conforme é possível verificar em [9]. Uma aplicação mais versátil pode ser implementada usando qualquer linguagem de programação e componentes de interface, nomeadamente as OWCs (que têm a vantagem de poderem ser carregadas com dados XML) ou Applets Java. Um exemplo de aplicações deste tipo são aquelas descritas em [18] e em [3], sendo, esta última, uma plataforma da categoria Thin-Client OLAP (Chris Harrington é, alias, um forte defensor de plataformas Thin-Client OLAP, mantendo um Web Blog sobre esse assunto, do qual este artigo faz parte). A plataforma analítica Report Portal [4] baseia-se no XMLA.

VII. CONCLUSÕES

Neste artigo foi feita uma análise comparativa, em termos de funcionalidades e limitações, entre as categorias de Soluções Client OLAP Fat-Client e Thin-Client. Cada uma destas categorias foi devidamente caracterizada, nomeadamente, em termos das situações às quais melhor se adequam. Assim, as Soluções Fat-Client OLAP apresentam-se como as mais adequadas às situações em que os utilizadores vão realizar processamento OLAP intensivo, nomeadamente: alterar a estrutura de Relatórios OLAP pré-definidos, através de Pivoting, ou através da alteração dos Membros e Dimensões que nele constem. Por outro lado, uma Solução Thin-CLient OLAP adequa-se melhor a utilizadores ocasionais, que não realizam processamento OLAP intensivo, por exemplo, só estão interessados em aceder a Relatórios OLAP pré-definidos sobre os quais muito raramente realizarão operações de Roll-Up, Drill-Down, Slice e Dice, e, provavelmente, nunca operações de Pivoting, nem alterarão os Membros das Dimensões que constam deles.

Seguidamente, foram analisadas uma série de Soluções Cliente OLAP, baseadas na Web, e suportadas pelo Analysis Services. Dessa forma, foi analisada uma Solução Fat-Client OLAP baseada nas OWCs, no PivotTable Services e na Plataforma .NET, e três Soluções Thin-Client OLAP, uma baseada na ferramenta Analysis Services Thin Web Client Browser, disponibilizada no MS SQL Server 2000 Resource Kit, uma outra baseada na utilização das OWCs, como objectos em memória, do lado do Servidor, e, finalmente, uma solução baseada no XMLA. Em relação a esta última Solução, é feita uma introdução e analise detalhada do Mecanismo de Acesso a Fontes de Dados Multidimensionais XMLA, onde são cobertos os seus Métodos e a sua implementação por parte do Analyis Services.

Fica agora a questão: qual é a Solução Client OLAP que melhor se adequa a uma determinada situação concreta? A escolha terá, certamente, de resultar de um compromisso entre a performance (em termos de tempo de resposta e funcionalidades analíticas) e a acessibilidade (em termos de tipo/configuração da plataforma de acesso e infra-estrutura de rede física/lógica utilizada) pretendida para o sistema. Paralelamente a esta análise de compromisso, há um outro aspecto relevante: as aplicações analíticas das organizações têm uma grande variedade de perfis de utilizadores, o que permite adaptar a arquitectura da solução disponibilizada ao perfil dos utilizadores. Neste sentido, é usual o maior grupo de utilizadores dos sistemas analíticos serem os menos exigentes, em termos de funcionalidades, podendo as suas necessidades serem satisfeitas através de relatórios estáticos ou parametrizados. Ao, frequentemente reduzido, grupo de utilizadores sofisticados, que requer, da aplicação, um alto desempenho e capacidades analíticas poderosas, a aplicação pode disponibilizar uma Arquitectura de Acesso Directo, limitada a ambientes Intranet ou VPN. Finalmente, ao grupo de utilizadores intermédio, que requer algumas capacidades exploratórias, mas, a liberdade dos acessos via Internet, pode-se disponibilizar acessos via XMLA ou HTTP/HTTPS.

REFERÊNCIAS

[1]: Bruney A. *The Microsoft Office Web Components Black Book with .NET*. Lulu Press. April, 2005.

[2]: "Tools, Samples, eBooks, and More". *SQL Server 2000 Resource Kit. Part 11. Chapter 39.* http://www.microsoft.com/technet/prodtechnol/sql/2000/reskit/part11/c3961.mspx (January 4, 2006).

[3]: Harrington C. "FLEXpart: Flexible OLAP charting with XMLA and OWC". *Active Interface. ThinOLAP Home.* December 2, 2004. http://www.activeinterface.com/b2004_12_2.html (January 4, 2006).

[4]: "Report Portal Homepage". *Report Portal.* http://www.reportportal.com/ (January 4, 2006).

[5]: "XML for Analysis Software Development Kit". *Microsoft Download Center*. Published: March 27, 2003. Revision: 1.0. http://www.microsoft.com/downloads/details.aspx?FamilyId=0E93B1EE-BE73-4D15-9C37-6CC2DCC2AC33&displaylang=en (January 2, 2006).

[6]: "XML for Analysis Specification Version 1.1". *XMLA HomePage. XMLA Documents.* http://www.xmla.org/docs_pub.asp (January 2, 2006).

[7]: Carvalhal C. "As MS Office Web Components e a sua Adequação ao Desenvolvimento de Aplicações Cliente OLAP". *Electrónica e Telecomunicações: Revista do Departamento de Electrónica e Telecomunicações*. Vol.4, N.º 6, March, 2006.

[8]: Carvalhal C. "Acesso ao Analysis Services via HTTP/HTTPS: Arquitecturas, Configurações e Mecanismos de Autenticação e Controlo de Acesso". *Electrónica e Telecomunicações: Revista do Departamento de Electrónica e Telecomunicações.* Vol.4, N.º 6, March, 2006.

[9]: Ericsson R. "XML for Analysis: Marrying OLAP and Web Services: Move your Analytical Applications to a Flexible Web-Based Architecture". *SQL Server Magazine.* InstantDoc #44006. November, 2004. http://www.windowsitpro.com/SQLServer/Article/ArticleID/44006/SQLServer_44006.html (April 12, 2005).

[10]: Whitney R. "XML for Analysis". *SQL Server Magazine.* InstantDoc #19846. April, 2001. http://www.windowsitpro.com/SQLServer/Article/ArticleID/19846/www.sqlmag.com/articles/d737/a19846 (April 12, 2005).

[11]: Hasan J., Tu K. "Build an OLAP Reporting App in ASP.NET Using SQL Server 2000 Analysis Services and Office XP". *MSDN Maganize*. October, 2003. http://msdn.microsoft.com/msdnmag/issues/03/10/OLAP/default.aspx (April 12, 2005).

[12]: Scott M., Lynn J. "Building a Web-Based Analysis System: A Real-World look at using the Analysis Services Thin Web Client Browser". *SQL Server Magazine.* InstantDoc #24692. June, 2002. http://www.windowsitpro.com/Article/ArticleID/24692/24692.html?Ad=1 (April 12, 2005).

[13]: Vishnubhotla S. "Using ADO MD and Office Web Components to Generate Thin Client Charts and PivotTables from OLAP Cubes". *ASP Today*. May, 2000. http://www.asptoday.com/Content.aspx ?id=353 (January 2, 2006).

[14]: Youness S. "Using MDX and ADO MD to Access OLAP Data with ASP". *ASP Today*. April, 2000. http://www.asptoday.com/Content.aspx?id=325 (January 2, 2006).

[15]: Youness S. "Using MDX and ADO MD to Access OLAP Data from ASP Part II". *ASP Today*. July, 2000. http://www.asptoday.com/Content.aspx?id=388 (January 2, 2006).

[16]: Pasumansky M. "Análisis Services 2005 Protocol –XMLA over TCP/IP". *SQLJunkies*. December 15, 2005. http://www.sqljunkies.com/Article/5412AA00-E109-40FB-8E6C-29FDEB05DC7C.scuk (January 2, 2006)

[17]: Rabeler C. "Communicating with Analysis Services". *SQL Server Magazine.* InstantDoc #46456. July, 2005. http://www.windowsitpro.com/SQLServer/Article/ArticleID/46456/46456.html (January 2, 2006).

[18]: Pasumansky M. "REX –Java OLAP Browser with XMLA support". *SQLJunkies. Mosha Pasumansky Web Blog.* http://sqljunkies.com/WebLog/mosha/comments/5228.aspx (April 12, 2005).