Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Luís Carlos Lopes Soares

Arquitectura Dual de Integrador de SIs Heterogéneos com Sincronização

Tese de Mestrado

Sistemas e Tecnologias de Informação para as Organizações

Professor Doutor Jorge Alexandre de Albuquerque Loureiro

Entidade Cooperante:

Visabeira Digital - Sistemas de Informação e Multimédia, S.A.

Outubro de 2011

Agradecimentos

Muitos foram aqueles que, directa ou indirectamente, colaboraram para a realização da presente dissertação. Expresso aqui o meu reconhecimento e a minha gratidão pela sua colaboração.

Primeiramente, quero agradecer ao Senhor Doutor Jorge Loureiro pelo cuidado e atenção com que me orientou ao longo do projecto desenvolvido e na redacção desta dissertação, tendo-se sempre demonstrado disponível para a solução de qualquer conflito que surgisse. Quero ainda agradecer o seu trabalho de revisão final desta dissertação e a confiança em mim depositada.

À Visabeira Digital – Sistemas de Informação e Multimédia, S.A., agradeço toda a disponibilidade e cooperação para a realização e aplicação deste projecto.

Ao Senhor Engenheiro António José Borges pelo seu companheirismo e facilitismo enquanto colaborador da Visabeira Digital.

Ao Senhor Hugo Henriques pela sua disponibilidade e prontidão para o esclarecimento de dúvidas relacionadas com a Viatel.

Por fim, não poderia esquecer todos aqueles que pelo permanente apoio, compreensão e amizade me motivaram a atingir este objectivo. Assim não poderia esquecer os meus familiares, em especial, os meus pais, e ainda os meus amigos mais próximos, Susana, Paulo e Pedro.

Obrigado a todos.

Resumo

Comunicação é sem dúvida a grande palavra-chave entre os sistemas informáticos. A criação de um canal de comunicação que permita a interligação de vários sistemas e fontes de dados, independentemente da sua localização física, poderá proporcionar um enorme valor acrescentado para a empresa.

Neste contexto, e com o objectivo de melhorar os seus processos de negócio, o Grupo Visabeira decidiu substituir o seu actual *Enterprise Resource Planning* (ERP), o GrVisa, pelo SAP.

Com a implementação do novo ERP surgiu a necessidade de desenvolver novas ferramentas de suporte à operação. A Viatel, empresa de telecomunicações do Grupo, alertou também para a necessidade de adquirir um sistema que permitisse a emissão de Guias de Transporte, mesmo na ausência de ligação ao servidor central. Este sistema deveria assegurar também a integração dos dados com o respectivo módulo do SAP.

Sendo a Visabeira Digital a empresa responsável pelo *software* do Grupo, analisou as vantagens que um sistema integrador poderia disponibilizar para outras empresas. Assim, pretendeu não só implementar a funcionalidade de emissão de Guias de Transporte, mas também estruturar uma arquitectura de interligação global que garantisse a expansão do sistema a novas funcionalidades e a outras empresas.

Como fruto da implementação deste projecto surgiu uma nova arquitectura, baseada na arquitectura cliente/servidor, estruturada e subdividida em três sistemas: Sistema Cliente, Sistema Integrador/Servidor e Sistemas de Informação Externa.

O Sistema Cliente implementa um funcionamento dual, permitindo que as suas funcionalidades possam aceder a dados em tempo real (modo *online*), operar recorrendo exclusivamente aos dados armazenados localmente (modo *offline*) e ainda, estabelecer um consenso entre ambos os modos de funcionamento.

O Sistema Integrador/Servidor foi criado com o objectivo de centralizar o acesso aos Sistemas de Informação Externos num único ponto. Desta forma, toda a comunicação e integração de dados entre os diversos Sistemas de Informação é processada por este.

Já os Sistemas de Informação Externos representam os Sistemas de Informação com os quais o Sistema Integrador/Servidor comunica para obter/integrar informação.

Após a implementação deste projecto os principais resultados obtidos foram: o aumento da produtividade; a integração automática de dados entre os diversos Sistemas de Informação; a redução de custos dos *links* entre armazéns; e, por fim, a criação de uma plataforma que permite a uniformização do desenvolvimento de *software*.

Palavras-chave: Arquitectura Dual, Sistema Online e Offline, SAP, Integração, Sincronização.

Abstract

Communication is undoubtedly the biggest keyword among informatics systems. The creation of a communication channel that allows the interconnection of various systems and data sources, regardless of their physical location, provides an enormous value to the company.

In this context, and with the objective of improving its business processes, the Visabeira Group decided to replace its existing Enterprise Resource Planning (ERP), the GrVisa, by SAP. With the implementation of new ERP, it was necessary to develop new tools to support the operation.

The Viatel, a telecommunications company of the Group, also warned about the need to acquire a system that would allow the issuance of waybills, even without a connection to the central server. This system should also ensure the integration of data with the respective SAP module.

Being Visabeira Digital the company responsible for the software of the group, it has analyzed the advantages that an integrator system can provide to other companies. The objective isn't just the implementation of the functionality of waybills issuing, but also to contribute to a global interconnection architecture that would guarantee the expansion of the system to new features and other companies.

As a result of this project implementation, a new architecture arises. This architecture, based on client/server architecture, was structured and divided into three systems: Client System, Integrator/Server System and External Information Systems.

The Client System implements a dual function, allowing their facilities to access data in real time (online mode), operate using only the locally stored data (offline) and also establish a consensus between these two operation modes.

The Integrator/Server System was created with the aim of centralizing access to external information systems at a single point. Thus, all communication and data integration between different Information Systems are processed by this system.

At last, the Externals Information Systems represents all Information Systems with which the Integrator/Server System establishes a communication for reading and integrating information.

After the implementation of this project, the main results were: increased productivity; automated data integration between various Information Systems; reduction of costs of the links between warehouses; and, finally, the creation of a platform that allows standardization of the software development.

Keywords: Dual Architecture, Online and Offline System, SAP, Integration, Synchronization.

Índice

Índice de ilustrações

Capítulo 1

Introdução

Com a globalização e o crescimento dos meios e tecnologias de comunicação, a tendência é de desenvolver aplicações para a Internet (*Web-Based*). Estas oferecem um acesso flexível, independente da sua localização física, sendo apenas necessário um acesso à rede (Intranet ou Internet) e um dispositivo (móvel ou não) com um dos diversos *browsers* existentes no mercado. A sua característica multiplataforma permite o acesso à aplicação *Web* independentemente do Sistema Operativo que o dispositivo possui. Nesta arquitectura, tanto a aplicação como o armazenamento de dados estão centrados na rede. Esta centralização é uma mais-valia quando pretendemos um sistema *online*. Isto porque, quer o acesso aos dados quer o acesso à aplicação, é partilhado e pode ser efectuado por um enorme número de utilizadores, independentemente da sua localização física ou do dispositivo através do qual é feito o acesso. De igual modo, qualquer alteração efectuada sobre os dados num sistema deste género tem repercussão imediata para todos os utilizadores. Por fim, uma outra mais-valia a considerar nesta arquitectura é a inexistência de manutenção do *software* disponível nas máquinas dos utilizadores [COULOURIS et. al., 2001].

Com a implementação de novos Sistemas de Informação no Grupo Visabeira [GrupoVisabeira, 2010], nomeadamente a substituição do ser ERP actual, surgiu uma nova necessidade. Esta necessidade é caracterizada pela falta de um Sistema de Informação que, embora enquadrado numa arquitectura cliente/servidor, fosse capaz de operar na ausência da ligação ao sistema servidor. Tipicamente um sistema *offline*. O presente projecto contempla assim, um estudo aprofundado sobre os sistemas *Web-Based* e sobre a arquitectura cliente/servidor com o objectivo de fundir as mais-valias de cada um deles num novo sistema. A este sistema reúne-se a característica *offline* atrás referida. A implementação de um novo sistema com funcionamento dual (*online* e *offline*) trará um enorme valor acrescentado para a empresa, aumentando a produtividade dos colaboradores, que poderia ser afectada por falhas de ligação entre os sistemas, e reduzindo os custos das ligações dedicadas, substituindo-as por alternativas mais económicas. Esta substituição é possível uma vez que a cache proporcionada pelo funcionamento *offline* do sistema, garante a operacionalidade de todo o sistema, mesmo através de ligações de dados mais lentas.

Outro factor não menos importante num Sistema de Informação é a expansão das suas funcionalidades. Assim, o novo sistema deverá ser expansível de modo a permitir a inclusão de novas funcionalidades completamente distintas, quer a nível de dados, quer a nível de lógica de negócio ou empresa para a qual actuam. Esta característica será uma mais-valia, uma vez que o Grupo Visabeira é constituído por um conjunto de empresas para as quais se espera vir a desenvolver funcionalidades no âmbito deste projecto.

O acesso dos utilizadores ao sistema deve ser configurável utilizador a utilizador, não só ao nível das funcionalidades a que este possui acesso, mas também ao nível dos acessos da organização (Firmas, Centros Logísticos, Armazéns, etc.).

Por outro lado, e uma vez que estamos a falar de um integrador de Sistemas de Informação Heterogéneos, este deverá garantir com sucesso a integração dos dados provenientes das diversas funcionalidades com os vários Sistemas de Informação Externos à arquitectura, nomeadamente o Sistema de Informação SAP ERP.

O estudo do novo sistema será fragmentado em quatro áreas distintas. Uma área de definição dos sistemas envolvidos. Uma área de integração com os diversos Sistemas de Informação externos à arquitectura, nomeadamente o SAP ERP [SAP, 2011]. Uma terceira área de definição das interfaces e canais de comunicação e por último, a própria aplicação informática.

Após desenvolvimento da arquitectura proposta neste projecto, a sua implementação no Grupo Visabeira será faseada. Numa primeira fase será instalada em alguns armazéns da empresa Viatel, estendendo-se depois aos restantes armazéns. Posteriormente, com o crescimento da solução apresentada e com a inclusão de novas funcionalidades, quer sejam elas da mesma área de negócio ou de áreas de negócio diferentes, a instalação será efectuada de acordo com as necessidades das respectivas empresas do grupo. No futuro, caso se verifique a aquisição ou o desenvolvimento de um novo Sistema de Informação e este necessite de obter ou integrar dados com os actuais Sistemas de Informação do Grupo Visabeira, o novo sistema permitirá a sua incorporação como Sistema de Informação Externo.

1.1 Enquadramento

O Grupo Visabeira, SGPS, S.A. nasceu em 1980. A sua sede localiza-se na cidade de Viseu, uma região entre o Dão e o Mondego, debruada mais ao longe pelas Serras da Estrela e do Caramulo, possuindo uma paisagem natural de rara beleza e um valioso património arquitectónico. Este é hoje uma holding multinacional organizada em cinco sub-holdings (Visabeira Global, Visabeira Indústria, Visabeira Turismo, Visabeira Imobiliária e Visabeira Participações Financeiras), com uma presença destacada em doze países, comercializando os seus produtos e serviços em mais de quatro dezenas de nações, nos cinco continentes.

Após vários anos a utilizar o seu próprio ERP, denominado de GrVisa, assistiu-se a uma progressiva dificuldade em acompanhar o crescimento e expansão do Grupo Visabeira. Dificuldades estas, provenientes da descentralização de alguns módulos funcionais, nomeadamente os módulos Financeiro e Logística. Outras dificuldades, não menos importantes, residiram no simples facto do GrVisa ter sido um alvo de sucessivos processos de manutenção e desenvolvimento, durante o qual não foi produzido qualquer tipo de documentação, que pudesse vir a auxiliar a equipa de manutenção e desenvolvimento de *software*. A ausência desta documentação levou a que cada programador, face ao desconhecimento do sistema, não reutilizasse devidamente o código e as entidades existentes, que acabariam mais tarde por ficar esquecidas. Assim, o processo de manutenção tornou-se cada vez mais complexo e ineficiente, comprometendo em alguns casos o desempenho e a informação armazenada no ERP. Desta forma, o Grupo Visabeira, impulsionado pelo seu carácter inovador, procedeu à aquisição e implementação de um novo ERP, o *Systems, Applications and Products* (SAP). O principal objectivo desta nova aquisição focou-se na centralização de informação, e na redefinição e melhoria dos processos de negócio existentes nas diversas empresas que constituem o Grupo.

Relativamente ao SAP, a sua linguagem de programação é o *Advanced Business Application Programming* (ABAP). Esta linguagem de quarta geração (4GL) foi criada para desenvolver o próprio SAP. Foi melhorada e é actualmente utilizada pelos implementadores de SAP no desenvolvimento das suas aplicações, que normalmente são extensões ou adaptações a requisitos mais específicos do negócio [KRETSCHMER e WEISS, 1996].

Porém, existem algumas dificuldades que advêm da implementação deste ERP. Uma delas é o facto do novo ERP possuir inúmeras transacções. Uma transacção SAP é descrita como uma sequência de passos relacionados com a área de negócio, quer a nível funcional quer a nível lógico [WILL et. al., 1998]. Torna-se assim um pouco difícil memorizar quais as transacções e a ordem que estas deverão ser executadas para cumprir determinado processo de negócio, tornando-o propício a erros. Assim sendo, cada processo traduz um acréscimo de responsabilidade por parte do utilizador, não só ao validar os dados introduzidos, mas também ao garantir que efectuou todas as transacções necessárias. Caso contrário, a informação apresentada pelo sistema poderá não reflectir a realidade [SAPMM, 2010].

Após a identificação de algumas dificuldades que advêm da utilização do novo ERP, rapidamente apercebeu-se da necessidade de desenvolver novas ferramentas de suporte à operação, com

o intuito de simplificar processos e complementar algumas carências a que o sistema SAP não era capaz de responder.

Para melhor compreender o problema a que este projecto procurou dar resposta, importa descrever a arquitectura e os módulos funcionais do sistema SAP, apresentados na Ilustração 1.

Ilustração 1 – Módulos funcionais do SAP.
(Fonte: [GRAVE, 2002])

Ao analisarmos a Ilustração 1 verificamos que a arquitectura do SAP está subdividida em três principais áreas/módulos funcionais: Logística (cor verde), Financeira (cor vermelha) e Recursos Humanos (cor amarela). Existe ainda um módulo de Fluxos de Trabalho (cor azul) que gere os fluxos de informação. Todos estes módulos são integrados e partilham a mesma origem de dados [WILL et. al., 1998].

O Grupo Visabeira, nesta primeira fase, adquiriu dois módulos da área funcional Logística, o módulo *Materials Management* (MM) e o módulo *Sales and Distribution* (SD). De igual forma, foram também adquiridos módulos da área funcional Financeira e de Recursos Humanos. Contudo, estes módulos não serão descritos detalhadamente, uma vez que o objecto do presente projecto não tem ligação com eles.

Como se pode ver na Ilustração 1 o módulo MM é totalmente integrado com os outros módulos do SAP ERP. Este dá suporte a todas as fases da gestão de materiais, desde o planeamento de produção, controlo de materiais, recepção de mercadorias, gestão de stock, até ao pagamento final ao fornecedor utilizando o módulo de verificação de facturas [WILL et. al., 1998].

Por outro lado, o módulo SD é uma parte da área de logística que dá suporte aos clientes através de cotações, pedidos de vendas e toda a análise de facturação do cliente. Permite ainda que a empresa introduza os seus preços de venda, verifique os pedidos em aberto e consiga fazer previsões de pagamentos [WILL et. al., 1998].

A implementação destes dois módulos está a ser realizada por etapas. Na primeira etapa, serão utilizados apenas por algumas empresas do Grupo, nomeadamente a Viatel, empresa de telecomunicações, detentora de cerca de 50 armazéns em todo o território nacional.

Com a implementação dos módulos MM e SD do SAP, a Viatel apercebeu-se o quanto estes iriam melhorar os seus processos internos. Contudo, o SAP não permite o seu funcionamento em modo *offline*. Desta forma, a emissão de Guias de Transporte, aquando falhas de ligação, não esta-

va garantida. Com este objectivo, a Viatel solicitou à Visabeira Digital o desenvolvimento de uma aplicação *offline* que permitisse a emissão e validação de Guias de Transporte pelo fiel de armazém e, posteriormente, a integração automática das mesmas com o respectivo módulo SAP.

O problema imediato seria garantir a disponibilidade de emissão de Guias de Transporte para a Viatel. Esta disponibilidade seria assegurada pela implementação dos modos *online* e *offline* na funcionalidade (subaplicação) de emissão de Guias de Transporte. A implementação dos dois modos anteriormente referidos seria uma vantagem na medida em que a Viatel teria ao seu dispor um sistema capaz de operar mesmo aquando falhas de ligação, reduzindo as quebras de produtividade inerentes a esta situação.

Contudo, pretendia-se que o sistema em estudo pudesse abranger todas as empresas do Grupo Visabeira, não só pela implementação do funcionamento dual (*online* e *offline*) nas funcionalidades existentes, mas também através da inclusão de novas funcionalidades distintas umas das outras, quer a nível de dados, quer a nível de lógica de negócio. Desta forma, o sistema permitirá a centralização de funcionalidades e do desenvolvimento de *software,* das diversas empresas do grupo, neste único sistema. Esta centralização traduzir-se-ia numa vantagem para a Visabeira Digital, isto porque, uma vez que a linguagem de programação seria única e do conhecimento de todos os programadores, estes poderiam partilhar o seu conhecimento e as soluções adoptadas em determinadas circunstâncias. De igual modo, pelo facto de o sistema agregar um conjunto de funcionalidades com a mesma estrutura de desenvolvimento, permitiria assim a diminuição do tempo necessário para o desenvolvimento/manutenção de *software*. Esta redução de tempo só seria possível uma vez que qualquer um dos programadores teria conhecimento da sua estrutura e estaria apto para efectuar todo o tipo intervenção. Outra vantagem da centralização de funcionalidades, agora numa vertente das restantes empresas do grupo, seria o facto de que cada empresa teria ao seu dispor uma área de trabalho representada por uma aplicação informática que agruparia e disponibilizaria devidamente todas as funcionalidades a que cada colaborador teria acesso.

De forma a facilitar a actualização e implementação de novas funcionalidades, o sistema deveria permitir a integração das mesmas através de módulos de *software*, tipicamente sob o formato de *Dynamic-Link Library* (DLL).

Para enquadrar melhor o leitor na problemática deste projecto, importa conhecer a origem do problema imediato, anteriormente identificado. Actualmente, os armazéns possuem *software,* desenvolvido pela Visabeira Digital, que está interligado com o actual ERP, o GrVisa. Para estabelecer uma ligação entre este *software* e os servidores onde está localizado o ERP, cada armazém dispõe de um *link* directo com a sede com custos médios de 2000€ por ano. Uma vez que só a Viatel é detentora de cerca de 50 armazéns, os *links* traduzem-se num custo anual de 100.000€. Este *link* nem sempre está operacional devido a falhas de serviço que podem ter várias origens, tais como: a localização física dos armazéns em zonas mais rurais, acontecimentos meteorológicos adversos, ou outro tipo de acidentes provenientes de causas naturais ou não. A indisponibilidade do *link* obriga a que os fiéis de armazém tenham que recorrer ao papel para registar temporariamente a informação dos materiais que dão entrada/saída do armazém. Este acto traduz-se num aumento de custos de economato, além do tempo despendido a registar os materiais em papel e

posteriormente efectuar o mesmo registo no sistema informático, para não falar de possíveis extravios das folhas de papel.

Assim, a aplicação solicitada pela Viatel eliminaria o registo temporário de informação em papel, efectuado aquando falhas de ligação com o sistema central. Situações de extravio de documentos também ficariam solucionadas, uma vez que o sistema estaria sempre disponível e o registo de documentos estaria garantido. A integração automática dos documentos, com o módulo SAP, permitiria a actualização de consumos e stocks, contribuindo significativamente para o aumento da eficácia e eficiência dos processos de negócio da Viatel.

A importância da emissão das Guias de Transporte prende-se com o facto de estas serem o documento legal emitido pelo transportador, para acompanhar a mercadoria durante o transporte em território português. Cada vez que um técnico se dirige a um cliente para efectuar uma intervenção (instalação ou manutenção de equipamentos) ou transfere materiais entre armazéns, este é obrigado a ter na sua posse uma Guia de Transporte, sob pena de ser autuado. A Guia de Transporte deve identificar discriminadamente todos os materiais e ferramentas que ela acompanha, bem como a origem e o destino dos mesmos. Esta informação deverá seguir as normas impostas pelo Decreto-lei n.º 147/2003 de 11 de Julho. Com o intuito de esclarecer algumas questões relativas à certificação de *software* que foram surgindo, contactou-se o corpo jurídico do Grupo Visabeira, o qual não identificou qualquer requisito legal além do Decreto-lei identificado anteriormente.

Por outro lado, quando o técnico ou algum dos armazéns da Viatel verifica que tem falhas de stock em produtos importantes para a sua actividade, estes poderão, a qualquer momento, efectuar um pedido de material ao armazém local, com o intuito de repor o stock que tem em falta ou em menor quantidade. Este processo está representado na ilustração seguinte.

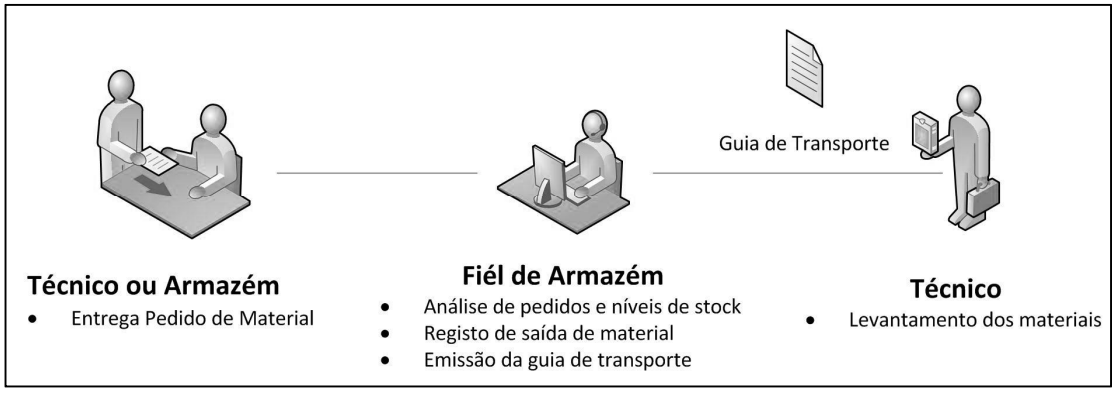

Ilustração 2 – Processo de pedidos de materiais.

Na Ilustração 2 verificamos que após a recepção do pedido de material, o fiel de armazém irá proceder à análise dos níveis de *stock*, recolher e registar a saída dos materiais solicitados e proceder à emissão da Guia de Transporte. O último passo, conforme mostra a Ilustração 2, é o levantamento da Guia de Transporte e dos respectivos materiais, pelo técnico. Durante este acto o técnico deverá assinar a guia.

A centralização dos dados provenientes das Guias de Transporte tem um enorme valor acrescentado, uma vez que esta permite, a qualquer momento, um rápido e eficaz confronto com o stock

das equipas técnicas e armazéns. Este confronto possibilita uma fácil detecção de casos de fraudes e/ou desvios de materiais.

Uma vez que a Visabeira Digital é a empresa responsável pelo desenvolvimento e manutenção de *software* e serviços informáticos das empresas do Grupo Visabeira, foi incumbida de analisar as vantagens que este tipo de sistema (*online* e *offline*) poderia disponibilizar também para outras empresas. Feito o estudo, concluiu-se que o funcionamento dual poderia ser uma mais-valia quando aplicado também a outras áreas de negócio. Assim sendo, pretende-se não só implementar a funcionalidade de emissão de Guias de Transporte para este caso específico, mas também, estruturar o sistema de modo a que a expansão do mesmo a novas funcionalidades e a outras empresas esteja assegurada. O presente projecto surge no seguimento da análise e levantamento de necessidades destas duas empresas. Para melhor compreendermos a situação, importa descrever o ambiente de Sistemas de Informação actual.

A solução actual do Grupo Visabeira conta com três principais Sistemas de Informação: o ERP desenvolvido pela Visabeira Digital de nome GrVisa, o sistema adquirido recentemente (SAP ERP) e um variado conjunto de aplicações *Web-Based*, desenvolvidas também pela Visabeira Digital sobre a plataforma SharePoint [MSSharePoint2010, 2010], designadas por BeOn. A ilustração seguinte indica de forma organizada as principais áreas funcionais de cada sistema acima referido.

Ilustração 3 – Sistemas de Informação do Grupo Visabeira.

O GrVisa é um ERP proprietário do Grupo Visabeira que foi desenvolvido por medida ao longo de 30 anos usando a linguagem de programação Delphi. A documentação do GrVisa é escassa ou mesmo inexistente, o que dificulta qualquer tentativa de manutenção ou crescimento das suas funcionalidades. Contudo, não será extinto para já. Muitas das empresas do grupo estão perfeitamente enquadradas e suportadas por este sistema de informação. Num futuro próximo, o GrVisa irá continuar a dar suporte a estas empresas até que módulos equivalentes, no Sistema de Informação SAP, sejam devidamente configurados e parametrizados.

Relativamente ao SAP ERP pretende-se rentabilizar a sua plataforma estável e de grande centralização. A possibilidade de expansão e integração de dados com os outros Sistemas de Informação, actualmente existentes no Grupo Visabeira, é outra mais-valia deste sistema. Por fim, e não menos importante, é a redefinição e melhorias de processos de negócio que este irá implementar.

Para finalizar, o último Sistema de Informação apresentado na Ilustração 3, são as aplicações BeOn. Estas aplicações estão subdivididas em duas categorias BeOn Office e BeOn Mobility. As aplicações BeOn Office implementam um conjunto de funcionalidades centrais à gestão de recur-

sos humanos, aos recursos partilhados e à administração interna. Por outro lado, as aplicações BeOn Mobility disponibilizam um conjunto de funcionalidades de suporte à mobilidade e vocacionadas para a área operacional. Isto é, vulgarmente são acedidas pelos técnicos através de um PDA e têm como principal finalidade disponibilizar em tempo real a informação proveniente das diversas acções dos técnicos (registo de produção, intervenções a clientes, etc.). Alguma da informação registada é lançada directamente no GrVisa e, recentemente, em SAP. Para que a informação armazenada nestes três sistemas seja coesa, é necessário manter processos rotineiros (*jobs*) de actualização e replicação de dados entre eles.

Apresentados os três Sistemas de Informação principais, torna-se importante perceber a forma como estes se interligam. De um modo geral, o acesso aos sistemas acima descritos pode ser interno (através da rede local) ou externo (através da Internet). A Ilustração 4 apresenta um esquema da arquitectura actual dos Sistemas de Informação do Grupo Visabeira, e a respectiva comunicação entre eles. Como podemos observar, todos os Sistemas de Informação estão inseridos na mesma rede de dados (rede interna do Grupo Visabeira) e comunicam entre si.

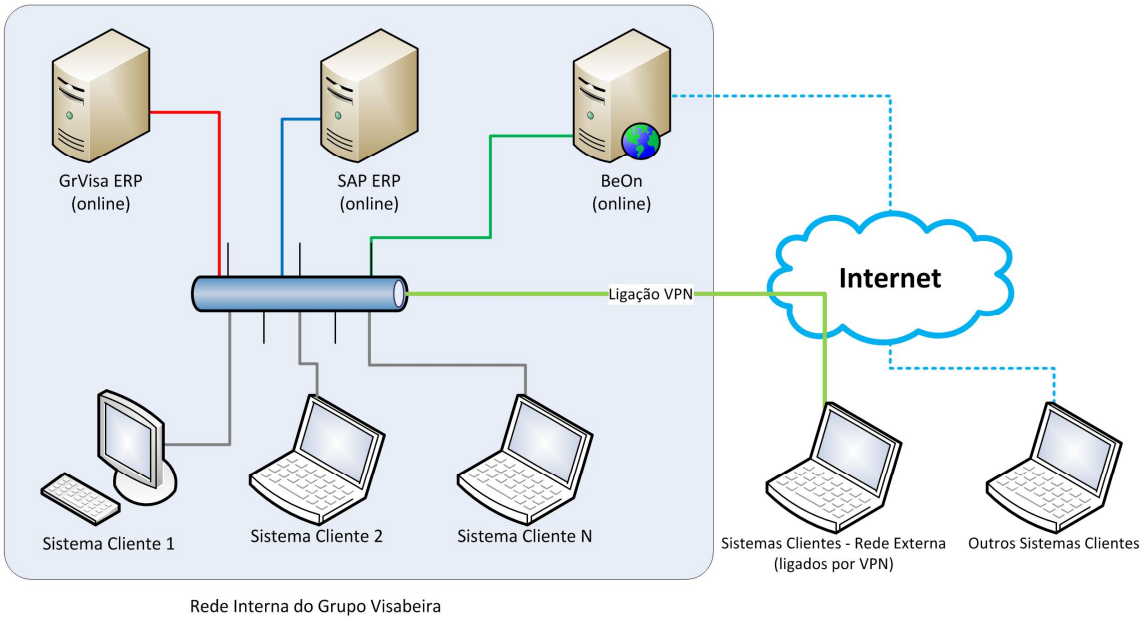

Ilustração 4 – Esquema da arquitectura actual.

Porém, e uma vez que algumas das empresas do Grupo Visabeira, nomeadamente a Viatel, subcontratam outras empresas às quais adjudicam obras ou tarefas que tem a seu encargo, surge a necessidade de efectuar um controlo junto destas empresas externas. Deste modo, para que seja possível avaliar o serviço prestado e proceder à respectiva facturação, surgem as aplicações *Web* englobadas no projecto BeOn.

Como podemos analisar na Ilustração 4, as aplicações BeOn encontram-se publicadas num servidor *Web*, permitindo o seu acesso quer pela rede interna (linha de cor verde), quer pela Internet (linha de cor azul). Estas aplicações são de fácil acesso, necessitando apenas de um *browser* para poderem ser utilizadas. O seu modo de operar divide-se em duas áreas distintas. Uma área de *front-office* destinada ao registo de produção por parte dos subcontratantes, e uma área de *back-office* onde são efectuados os processos de validação, avaliação dos serviços prestados e respectiva

facturação. As aplicações BeOn apresentam uma arquitectura própria que permite o balanceamento de carga de forma a optimizar as soluções que este disponibiliza. Esta arquitectura possui três camadas estruturadas de acordo com o esquema apresentado na Ilustração 5.

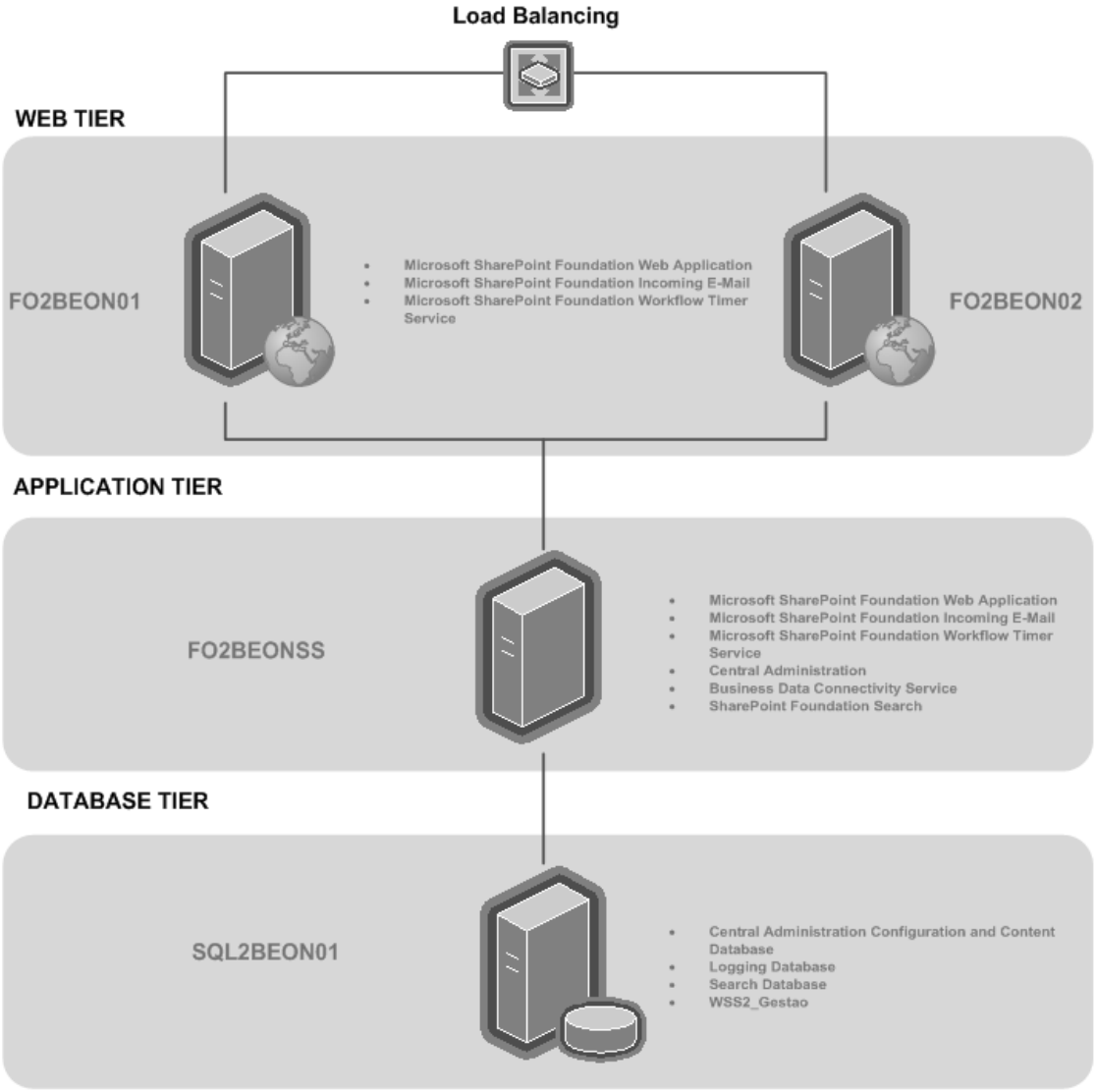

Ilustração 5 – Arquitectura das soluções BeOn.
(Fonte: [SI BeOn 2010])

A camada *Web* é a responsável por atender a pedidos de acesso. Para optimizar a sua capacidade de resposta, esta camada implementa dois servidores de iguais características e configurações. A camada de aplicação é onde está implementada a área de administração e configuração deste sistema. Por fim, a camada de Base de Dados, onde se encontra o servidor responsável por armazenar, entre outros, as Bases de Dados de cada *site*.

Por outro lado, as aplicações do GrVisa têm que obrigatoriamente ser acedidas através da rede interna, conforme mostra a Ilustração 6.

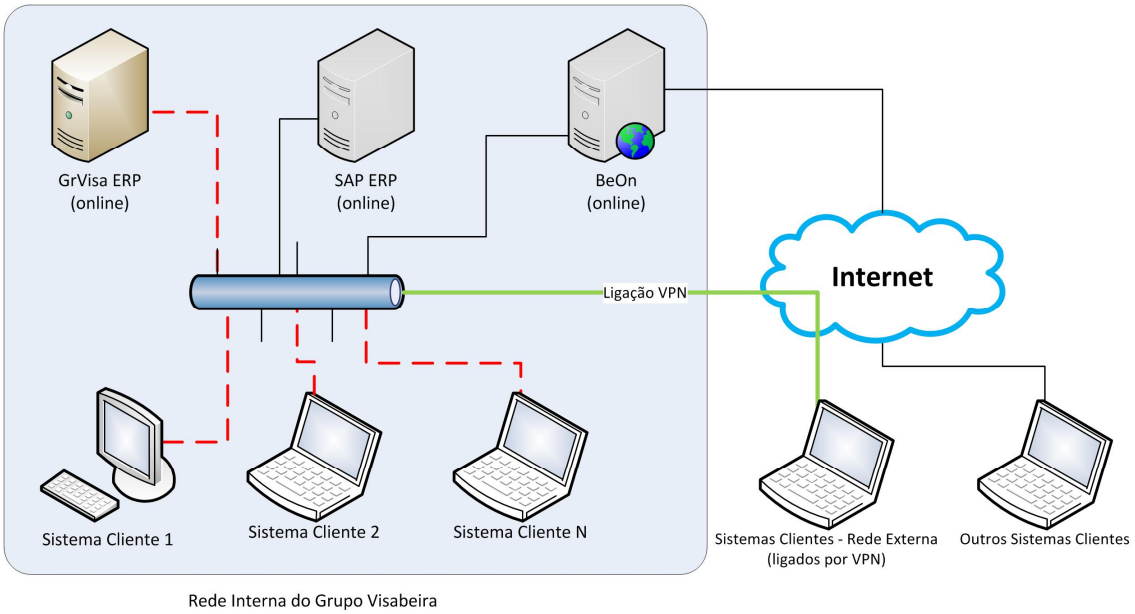

Ilustração 6 – Esquema de comunicação das aplicações GrVisa.

Cada aplicação pertencente ao GrVisa estabelece uma ligação TCP/IP ao servidor através de um Endereço IP e uma Porta previamente configurada. Como política de segurança a Visabeira Digital limitou o acesso a estas portas apenas à rede interna do Grupo Visabeira. Face à distância física e à fraca ligação de dados com os vários armazéns, esta limitação tornou-se uma barreira. Para tal, foi necessário efectuar contratos de fornecimento de *links* dedicados entre a sede do Grupo Visabeira e os armazéns, de forma a permitir a criação de ligações VPN com o mínimo de velocidade e qualidade exigida, o que levou aos custos bastante elevados atrás referidos. Estas ligações estão representadas a cor verde, na ilustração acima apresentada.

Analisando a arquitectura actual e a estrutura de comunicação entre os diversos Sistemas de Informação do Grupo Visabeira, apresentadas anteriormente, facilmente se detectam algumas limitações, anomalias e até mesmo alguns problemas. Um dos principais problemas surge com a diversidade de canais de comunicação existentes e definidos nas diversas aplicações clientes. Por exemplo, caso uma determinada funcionalidade necessite de informação proveniente dos três principais Sistemas de Informação (GrVisa, SAP e BeOn), esta terá que implementar os diversos métodos para aceder aos dados de cada um dos sistemas referidos, assim como definir todas as regras de validação e a lógica de negócio a aplicar sobre os dados. Inerente a este tipo de desenvolvimento está a dificuldade de manutenção, quer dos métodos de acesso a dados, caso exista a necessidade de os alterar, como por exemplo devido a migrações dos Sistemas de Informação para outros servidores, quer das regras de validação dos dados. Com o objectivo de ultrapassar este problema, a solução proposta neste projecto adiciona um Sistema Integrador à arquitectura actual. Este Sistema Integrador será o responsável por centralizar o acesso e interligar os três principais Sistemas de Informação envolvidos neste projecto, conforme apresenta o esquema da Ilustração 7.

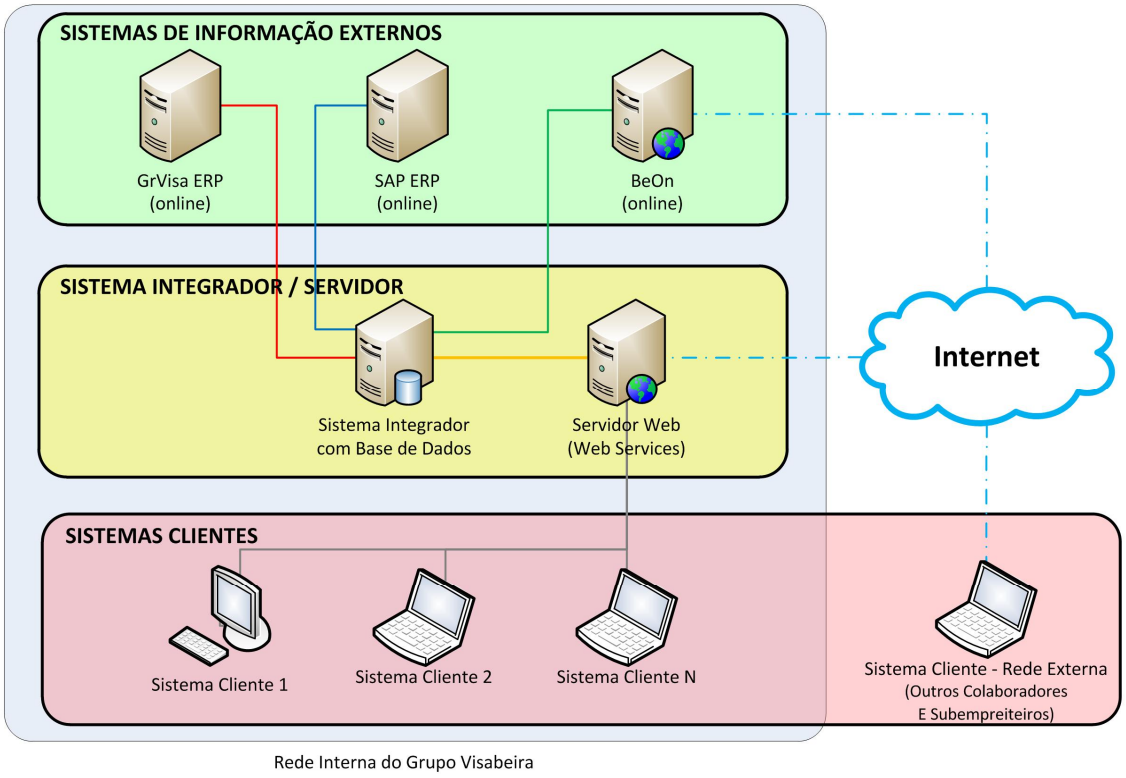

Ilustração 7 – Esquema com a arquitectura proposta neste projecto.

A vantagem deste integrador central é, como o próprio nome indica, a centralização e partilha de acesso aos diversos Sistemas de Informação da organização. Veja-se o paralelo entre as duas soluções: a actual e a proposta. Segundo o esquema actual (Ilustração 4), cada aplicação cliente estabelece uma ligação directa a um sistema e é esta que determina o modo como acede ao sistema, bem como implementa as regras e a lógica do respectivo negócio. Uma vez que existem funcionalidades semelhantes quer em aplicações *desktop* quer em aplicações *Web*, esta situação traduz-se na replicação da validação dos dados e da lógica de negócio, o que aumenta em muito a dificuldade na alteração e manutenção de ambas as funcionalidades. Para complicar um pouco a situação, surge o facto de a linguagem de programação não ser a mesma. Na solução proposta (Ilustração 7) este problema não existirá, uma vez que a interface que implementa as regras e a lógica de cada negócio é única, partilhada e encontra-se no Sistema Integrador, não em cada aplicação cliente.

Observando a Ilustração 7, vê-se que o Sistema Integrador dispõe de uma Base de Dados. Analise-se a sua função. A Base de Dados que o Sistema Integrador deverá contemplar tem o objectivo de garantir o funcionamento das aplicações clientes, mesmo quando os restantes Sistemas de Informação principais não estejam operacionais. Contudo, apenas é garantida a operacionalidade das funcionalidades capazes de operar no modo *offline*. Esta Base de Dados poderá ainda servir para disponibilizar e partilhar alguma informação entre Sistemas de Informação, funcionando como uma espécie de cache (armazenamento temporário), reduzindo assim o número de pedidos efectuados aos Sistemas de Informação. Por exemplo, os materiais associados a um determinado centro logístico variam pouco, assim em vez de as aplicações clientes e outros Sistemas de Informação

consultarem constantemente o sistema SAP (sistema onde estes são geridos), poderão obter essa informação da Base de Dados local do Sistema Integrador, que será actualizada regularmente.

Analisando novamente a Ilustração 7, verificamos que todas as aplicações de cada Sistema Cliente acedem ao Sistema Integrador unicamente através de um servidor *Web*, abolindo-se assim a ligação directa entre sistemas. Este servidor, recorrendo a *Web Services*, estabelece uma ponte entre as máquinas clientes e o Sistema Integrador/Servidor responsável pelo encaminhamento de pedidos para o respectivo Sistema de Informação. Nas situações em que seja necessário migrar algum Sistema de Informação para outro servidor, apenas será necessário configurar o Sistema Integrador com a nova ligação e não todas as aplicações clientes, uma vez que estas comunicam com o Sistema de Informação Externo através do Sistema Integrador.

Um outro obstáculo identificado na arquitectura actual, apresentada na Ilustração 6, prende-se com o facto de todas as aplicações interligadas com o GrVisa apenas funcionarem quando o acesso é realizado através da rede interna. Caso o acesso seja estabelecido do exterior da organização, é necessário estabelecer antecipadamente uma ligação VPN, o que, no caso dos armazéns, se traduz num custo elevado proveniente da contratação de *links* dedicados anteriormente referidos. Na arquitectura proposta neste projecto, a comunicação dos Sistemas Clientes com o GrVisa é efectuada através do servidor *Web*, parte integrante do Sistema Integrador, que está localizado na mesma rede (rede interna). Assim sendo, os acessos aplicacionais das máquinas clientes aos sistemas principais (GrVisa, SAP ERP e BeOn) serão mais flexíveis e menos dispendiosos, isto porque o seu acesso poderá ser feito através da rede local ou unicamente pela Internet, conforme apresenta a Ilustração 7. Deste modo, o servidor *Web* permite a substituição dos *links* dedicados, necessários nas aplicações que requerem endereço da rede interna, por simples ligações ADSL ou 3G. Esta substituição é possível uma vez que o servidor *Web* possui múltiplas interfaces de rede, umas na rede interna outras na rede externa (Internet). Desta forma, as máquinas clientes podem, através da Internet, enviar a informação para o servidor *Web*. Esta informação pode ser integrada com o(s) respectivo(s) Sistema(s) de Informação de dois modos distintos (modo síncrono ou modo assíncrono). No modo síncrono a integração da informação é *on-time*, ou seja, no momento em que a informação é recebida pelo Sistema Integrador. Já no modo assíncrono, o Sistema Integrador armazena a informação recebida na sua Base de Dados local, e posteriormente envia-a para o respectivo Sistema de Informação. Caso as funcionalidades possam operar no modo *offline*, não é necessário manter a ligação de Internet sempre activa, uma vez que o Sistema Cliente permite que a informação seja sincronizada com o Sistema Integrador ao longo do tempo.

Por outro lado, as aplicações BeOn podem ainda ser acedidas directamente pela Internet. Contudo, e mais uma vez, será o Sistema Integrador o responsável pela aplicação das regras e lógica de negócio, no caso em que exista a necessidade de integrar dados introduzidos directamente nas aplicações BeOn com os outros Sistemas de Informação. A ilustração seguinte mostra o processo acima descrito recorrendo ao exemplo da criação de um Pedido de Férias através de uma aplicação BeOn.

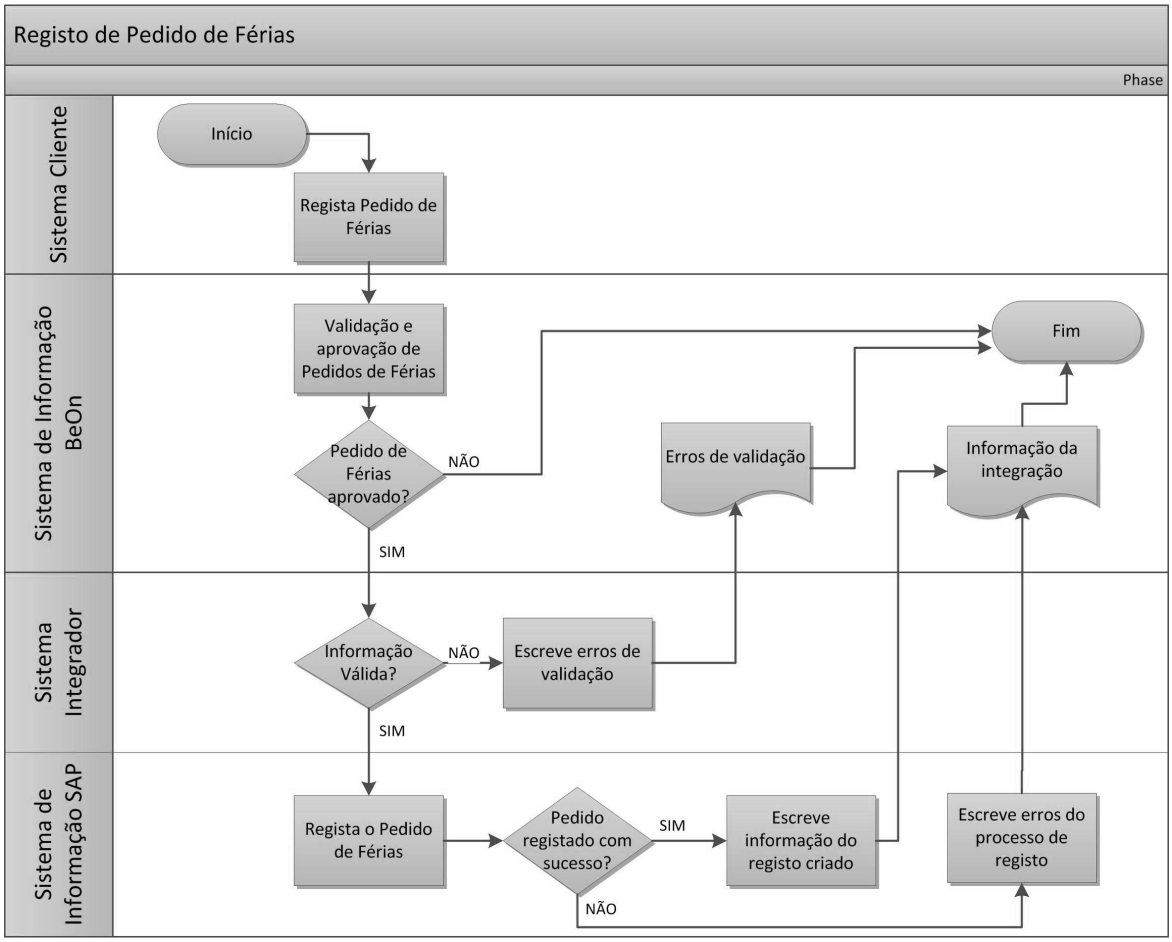

Ilustração 8 – Exemplo do registo de um pedido de férias.

Como podemos ver na Ilustração 8, o primeiro passo representa o acesso à aplicação BeOn por parte do colaborador e o registo do respectivo pedido de férias. Após criação com sucesso do pedido de férias segue o processo de validação e aprovação pelo seu responsável. Este processo é efectuado na actual aplicação BeOn. Após a aprovação do pedido de férias, a aplicação BeOn invoca o Sistema Integrador enviando a informação relativa ao pedido de férias aprovado. Por sua vez, o Sistema Integrador recebe e valida a informação do mesmo. Se porventura a informação do pedido de férias não for válida, o Sistema Integrador devolve os erros de validação para que posteriormente sejam corrigidos na aplicação BeOn. Caso toda a informação do pedido de férias seja válida, o Sistema Integrador avança com o processo de integração do registo no respectivo módulo de Recursos Humanos do SAP. Analisando a Ilustração 8 podemos observar que o resultado de todo o processo de integração (número do documento gerado em SAP, informação de sucesso/erro da operação, data e hora da integração, entre outros) é devolvido e armazenado em campos específicos na aplicação BeOn, interligando assim ambos os Sistemas de Informação.

Uma outra dificuldade encontrada na arquitectura actual assenta na gestão das rotinas (*jobs*) de actualização e replicação de dados entre os principais Sistemas de Informação, anteriormente enumerados. A nova arquitectura permitirá a centralização de todas as rotinas espalhadas pelos vários Sistemas de Informação, num único ponto, o Sistema Integrador/Servidor. Estas rotinas são as responsáveis por replicar, actualizar e integrar a informação entre os três principais Sistemas de Informação, anteriormente apresentados. Esta centralização facilitará a manutenção das rotinas e

permitirá escalonar os seus agendamentos para que estes não coincidam e sobrecarreguem demasiado os servidores num determinado momento.

1.2 Objectivos e motivações da investigação

Da discussão havida na secção anterior percebe-se que este projecto, tendo a sua génese na solução de um problema particular (a questão das Guias de Transporte a emitir pelos armazéns), rapidamente se expandiu, pois foram detectados muitos outros problemas e lacunas na arquitectura actual dos Sistemas de Informação do Grupo Visabeira. Analisando as dificuldades e problemas abordados na secção anterior, a implementação de uma nova arquitectura torna-se conveniente. Com a implementação desta nova arquitectura, não só se pensou na sua expansão em termos de escala (exportação da solução implementada para as Guias de Transporte a outras empresas do grupo), mas acabou-se também por pensar numa arquitectura integradora global, capaz de fornecer um conjunto de novas funcionalidades para áreas de negócio completamente distintas. Assim, o subsistema original acaba por ficar integrado neste novo sistema integrador. O alcance desta nova arquitectura tem intuitos bem mais extensos, proporcionados pela implementação de um sistema de informação dual. Isto é, um sistema de informação capaz de possibilitar o acesso a dados em tempo real (*online*), bem como permitir operações recorrendo exclusivamente aos dados armazenados localmente, sem necessidade de estabelecer uma ligação permanente ao sistema central (*offline*). Na mudança de estado, *offline* para *online*, o sistema deverá integrar de forma automática os dados entre os vários Sistemas de Informação existentes.

Tendo em conta o enquadramento feito anteriormente, podemos estruturar os objectivos em duas frentes: uma sobre a perspectiva da empresa de telecomunicações Viatel e outra, na perspectiva da empresa Visabeira Digital.

Segundo a Viatel, a importância do projecto, no imediato, é centrada nas Guias de Transporte e na possibilidade da sua emissão a qualquer instante. Assim, pretende dispor de um *software* capaz de satisfazer um conjunto de necessidades, tais como:

- Ser capaz de funcionar na ausência de uma ligação permanente ao sistema central (modo *offline*), como é o caso da funcionalidade de emissão de Guias de Transporte. Espera-se que, pelo facto do novo sistema estar sempre disponível, os níveis de produtividade dos técnicos aumentem, uma vez que o fluir do seu processo produtivo já não está limitado pela disponibilidade do *link* para os sistemas principais;
- Permitir que a funcionalidade Emissão de Guias de Transporte recolha a assinatura em formato digital através de um dispositivo. Este processo evita a impressão de documentos tipicamente impressos só para controlo. Um caso prático desta situação acontece quando surge uma urgência por falta de material. Assim sendo, os técnicos mais próximos, em vez de se dirigirem ao armazém para levantar material, encontram-se e trocam materiais entre si. Contudo, para que a troca de material seja registada no sistema, posteriormente, terão que emitir uma Guia de Transporte com o movimento entre os técnicos. Uma vez que a troca de material já foi efectuada e provavelmente o material já se encontra aplicado em obra, não é necessária a impressão da Guia de

Transporte, sendo esta apenas assinada digitalmente. Também a impressão de Guias de Transporte em duplicado será extinta. Isto porque, uma vez que a guia já está devidamente assinada e no sistema informático, apenas é necessária a impressão da via "Original" que, obrigatoriamente, acompanhará o técnico e será apresentada às autoridades, se assim o for solicitado;

- Disponibilizar uma funcionalidade que, no modo *online,* permita a gestão e validação das Guias de Transporte emitidas;
- Permitir a gestão dos acessos às funcionalidades, por utilizador, como por exemplo, permitir que cada utilizador apenas tenha acesso a determinados dados, nomeadamente a materiais e armazéns, i.e., apenas possa emitir guias de transporte de e para os armazéns a que tem acesso;
- Possuir um canal de comunicação que permita enviar alertas/notificações para os técnicos, alertando-os de situações específicas como por exemplo, informar o técnico do dia em que deve efectuar o inventário do armazém, ou simplesmente enviar uma mensagem colaborativa para todos os técnicos;
- Integrar, de forma automática, as guias de transporte com sistemas externos, nomeadamente o SAP MM, facilitando todo o moroso processo e ultrapassando a validação requerida pelas diversas transacções.

Já a Visabeira Digital pretende alargar o âmbito deste sistema, colocando a ênfase no sistema integrador, identificando como principais características, as seguintes:

- Permitir a expansão do número de funcionalidades disponíveis através de módulos de *software*. Numa primeira fase será a funcionalidade de emissão de guias de transporte. Contudo, pretende-se que outras empresas utilizem o mesmo sistema incluindo funcionalidades completamente distintas, como por exemplo, o registo do boletim de férias, onde os colaboradores acedem para poderem registar os seus pedidos de férias. Pretende-se com isto que a aplicação base não tenha dependência de nenhuma funcionalidade, reduzindo o tamanho da mesma e os recursos por ela ocupados. Desta forma, as funcionalidades serão incluídas como módulos de *software*, tornando o lançamento de novas versões mais simples;
- Uma vez que as funcionalidades serão distintas umas das outras, os dados que estas necessitam para operar poderão variar imenso. Por exemplo, se falarmos da funcionalidade emissão de Guias de Transporte, esta necessita da informação dos diversos armazéns, dos materiais, dos técnicos, entre outros. Já uma funcionalidade de facturação, por exemplo, necessita de um conjunto de informação completamente distinto (Clientes, Taxas de I.V.A., etc.). Com esta análise, o sistema deverá ser capaz de armazenar todos os dados necessários ao funcionamento das diversas funcionalidades, quer no modo *online* quer no modo *offline*;
- Dispor de dois estágios: Desenvolvimento/Qualidade e Produção. Estes estágios são importantes na medida em que permitirá um desenvolvimento seguro e eficiente, sem pôr em risco o sistema que está em produção. Por outras palavras, o estágio de Desenvolvimento/Qualidade terá como objectivo permitir o teste exaustivo de funcionalida-

des, enquanto o estágio de Produção tem o carácter final, onde as funcionalidades desenvolvidas e devidamente testadas ficarão acessíveis aos utilizadores;

- Desenvolver, nesta primeira implementação, um conjunto de automatismos e ferramentas, diminuindo assim o tempo e custos necessários ao desenvolvimento de novas funcionalidades;
- Permitir a fácil migração de aplicações já existentes (BeOn e GrVisa) para a nova plataforma;
- Dispor de um mecanismo de segurança capaz de bloquear o acesso à aplicação, quando esta não possua autenticação no servidor durante um determinado período de tempo, devidamente configurado. Considera-se este período o tempo máximo que uma máquina poderá emitir guias sem ter efectuado a sua autenticação no servidor, i.e., funcionar no modo *offline*. Existem dois motivos principais para a implementação deste mecanismo. O primeiro prende-se com o facto de as Guias de Transporte ficarem presas nas máquinas clientes, ou seja, não serem regularmente sincronizadas podendo gerar outros erros de integração de guias. Por exemplo, se um material seriado foi transferido de um armazém para outro, enquanto não for integrada a Guia de Transporte o armazém de destino não poderá dar entrada do respectivo material. Sendo obrigatória a autenticação *online* e regular da aplicação, para que esta seja desbloqueada, este problema estaria resolvido. O segundo motivo deve-se ao facto de, no modo *offline*, a autenticação dos utilizadores ser validada localmente. Assim, se porventura tenha sido retirado o acesso à aplicação a um determinado utilizador, este, mesmo não estando habilitados para tal, continuaria a poder usufruir da aplicação, desde que operasse no modo *offline*;
- Ser capaz de apresentar as funcionalidades em diversos idiomas, principalmente nos idiomas das empresas multinacionais do Grupo Visabeira;
- Permitir a integração de dados das várias funcionalidades com os diversos sistemas externos, de forma automática. No caso das Guias de Transporte, estas deverão ser integradas de forma automática com os respectivos módulos MM e SD do SAP, de forma a fazer a correcta movimentação de materiais e respectiva facturação, se for o caso;
- Reduzir o número de licenças necessárias para utilizar o SAP ERP. O SAP ERP permite um licenciamento personalizado. No caso do Grupo Visabeira, o número de licenças traduz-se no número de utilizadores ligados ao sistema. Contudo, o seu licenciamento varia de acordo com o seu carácter de acesso, i.e., se o acesso é feito de modo síncrono ou assíncrono. O novo sistema, ao contemplar um mecanismo de integração, apenas utilizará uma única licença (um utilizador) em modo assíncrono, uma vez que todos os outros utilizadores não actuam directamente em SAP, salvo algum acontecimento esporádico. Este mecanismo irá reduzir significativamente os custos de licenciamento;
- E, por último, reduzir os custos com os contratos de *links* dedicados entre o sistema central e os armazéns. A redução de custo poderá ser feita através da substituição dos *links* dedicados por alternativas mais económicas, tais como placas 3G. Esta substitui-

ção é possível face ao comportamento dual que o sistema permite, i.e., consegue operar com ou sem ligação à rede (modo *online* ou *offline*);

Todos os objectivos acima descritos foram identificados em reuniões onde participaram "*key-users*" e responsáveis das empresas Viatel e Visabeira Digital. A estes, acresce a motivação pela satisfação dos utilizadores. Para tal, serão abordadas *guidelines* de usabilidade no desenvolvimento das funcionalidades. Os objectos que estas irão conter deverão ter um *look-and-feel* próximo da solução Microsoft Office com o objectivo de proporcionar aos utilizadores uma maior proximidade com a aplicação, face ao conhecimento já adquirido nas ferramentas Office. Por outro lado, uma vez que as funcionalidades irão ser desenvolvidas à medida, estes não terão que memorizar/utilizar as complexas transacções do novo ERP SAP.

1.3 Estrutura da dissertação

Este projecto tem como objecto uma arquitectura global que implementa um sistema integrador de vários Sistemas de Informação heterogéneos. Esta arquitectura permite que um conjunto de aplicações informáticas seja integrado através de módulos de *software* e operem segundo um funcionamento dual (modo *online* e/ou *offline*). Não menos importante é a centralização do acesso aos Sistemas de Informação e das regras e lógica de negócio. Importou, portanto, enquadrar o leitor na problemática a ser tratada, mostrando as suas vertentes, os seus desafios, objectivos e motivos da investigação.

Depois desta introdução, inicia-se o capítulo 2 que apresenta as bases teóricas deste trabalho, fornecendo ao leitor uma visão geral do estado da arte e de possíveis sistemas que permitem alcançar, total ou parcialmente, alguns objectivos deste projecto.

O terceiro capítulo apresenta o estudo e a proposta da nova arquitectura, introduzindo os sistemas envolvidos, as principais características, interfaces adoptadas, bem como os canais de comunicação entre estes sistemas.

Com o quarto capítulo, descreve-se a fase de implementação da nova arquitectura. Entre outros assuntos, mostra-se a análise comparativa efectuada entre as várias tecnologias e ferramentas adoptadas. Esta análise encontra-se estruturada e dividida de acordo com os sistemas apresentados na arquitectura proposta. Como tal, tiveram-se em conta todos os requisitos impostos no projecto, quer pela Viatel, quer pela Visabeira Digital. O quarto capítulo apresenta também a aplicação informática, fruto da implementação da nova arquitectura. Por fim, enumeram-se alguns dos problemas que surgiram na implementação desta solução.

Por último, o quinto capítulo. Este contém as considerações finais do trabalho com as principais conclusões, contribuições e orientações para possíveis trabalhos futuros.

Capítulo 2

O Estado da Arte

O presente capítulo tem como objectivo apresentar um estudo geral do estado da arte e de possíveis sistemas capazes de alcançar e implementar, total ou parcialmente, alguns dos objectivos descritos neste projecto.

Realmente, como iremos ver, nos dias de hoje o mercado oferece-nos uma grande diversidade de soluções que, em parte, poderiam ser enquadradas e utilizadas neste projecto.

Efectuou-se assim uma pesquisa com a finalidade de verificar a existência de aplicações capazes de solucionar o problema em estudo, bem como, identificar soluções e ferramentas que permitissem a execução deste projecto. Atendendo aos objectivos atrás definidos, estas devem:

- Permitir a expansão da aplicação através do desenvolvimento e integração de pequenos módulos de *software*, uma vez que o desenvolvimento de novas funcionalidades será faseado;
- Disponibilizar algum controlo sobre a sincronização de dados entre as máquinas clientes e máquinas servidores;
- Possuir uma estrutura dinâmica de dados, que seja capaz de armazenar e sincronizar a diversidade de dados que cada funcionalidade necessita para operar. Este ponto será o responsável por garantir que, qualquer que seja a funcionalidade a desenvolver, seja possível enviar para o cliente toda a informação que ele necessite para operar no modo *offline*;
- Possibilitar a configuração de acessos ao utilizador, quer em termos de funcionalidades disponibilizadas quer relativamente aos dados a que este tem acesso.

Do fruto desta pesquisa destacaram-se duas soluções comerciais: Microsoft SharePoint e SalesForce.

2.1 Microsoft SharePoint

Uma vez que o desenvolvimento de *software*, o suporte ao armazenamento de dados e os sistemas operativos da Visabeira Digital são baseados em tecnologias e produtos Microsoft, a primeira pesquisa foi sobre o produto Microsoft SharePoint. Este enquadra-se na perspectiva do Sistema Integrador/Servidor que armazena e gere toda a informação. Para satisfazer as necessidades do Sistema Cliente, existe uma aplicação gratuita denominada de Microsoft SharePoint Workspace [MSSharePointWorkspace, 2010] que permite estabelecer uma ligação a *sites* do Microsoft SharePoint e posteriormente sincronizar e trabalhar dados em modo *offline*. Comecemos por apresentar o Microsoft Sharepoint.

O Microsoft SharePoint opera sobre o motor de base de dados MS SQL Server e um servidor Web, o *Internet Information Services* (IIS). Este pode ser instalado num único servidor, ou em *farm* [SharePointFarm, 2011], para um maior rendimento e fiabilidade. A sua instalação em *farm* não é nada mais que um conjunto de máquinas a trabalhar para um todo, a *farm*, aumentando assim a sua capacidade de resposta. Este tipo de instalação permite obter um sistema escalável uma vez que este possibilita a adição de novas máquinas à *farm,* de acordo com as necessidades. De igual modo, é um sistema mais robusto, uma vez que a *farm* permite a redundância e disponibilidade das aplicações contidas no sistema. A redundância refere-se ao uso de vários servidores, com iguais configurações, num ambiente com balanceamento de carga. Neste cenário, o objectivo é o de melhorar o desempenho da *farm*, mesmo que o número de utilizadores aumente consideravelmente. Já a disponibilidade é um conceito mais especializado que tem como referência um ambiente de vários servidores. Este ambiente é projectado para aceitar ligações e funcionar normalmente mesmo quando um ou mais dos servidores da *farm* não estão operacionais. Portanto, a disponibilidade implica redundância [CALLAHAN, 2008].

O funcionamento do SharePoint é muito simples e semelhante a outros gestores de conteúdos. Para a criação de *sites*, o SharePoint disponibiliza um conjunto completo de ferramentas que podem ser utilizadas na criação de qualquer tipo de site (*wikis*, repositório de documentos, sites para parceiros e clientes, entre outros). Disponibiliza ainda, uma infra-estrutura denominada de "*Central Administration*", que permite a fácil administração de todos os *sites*. Este fornece também um conjunto de componentes *Web* prontos a utilizar e que podem ser facilmente incorporados em páginas personalizadas [NOEL e SPENCE, 2007]. Como podemos ver na Ilustração 9, o seu ambiente gráfico, com o friso principal devidamente agrupado, é muito semelhante ao Microsoft Office, o que permite uma aproximação rápida dos utilizadores face ao conhecimento já adquirido.

Ilustração 9 – Aspecto gráfico do SharePoint 2010.

O SharePoint armazena a informação no formato de listas. Estas listas, à semelhança das tabelas no modelo relacional, possuem campos com os respectivos tipos de dados. A Ilustração 10 mostra a interface do SharePoint que permite a criação de um novo campo (coluna), neste caso do tipo "Data" e de preenchimento não obrigatório.

Ilustração 10 – Funcionalidade SharePoint para criação de novas colunas.

O SharePoint disponibiliza um conjunto alargado de tipos de listas (contactos, calendário, tarefas, hiperligações, inquéritos, anúncios, debates, entre outros) com algumas funcionalidades já associadas. Por exemplo, ao criar-se uma lista do tipo "tarefas de projectos", para além de um con-

junto base de colunas, tem ainda associada uma lista de tarefas e um mapa de *Gant* sobre as respectivas tarefas, como se pode ver na ilustração seguinte.

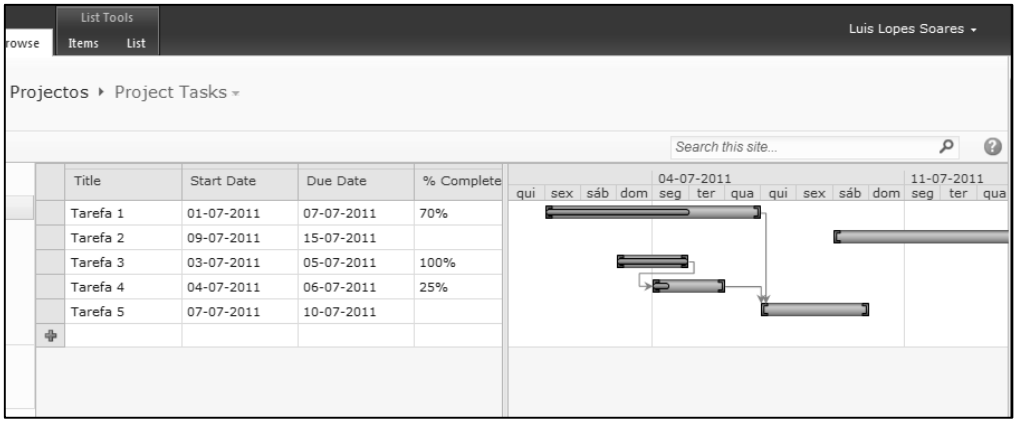

Ilustração 11 – Lista de SharePoint para organizar tarefas de um projecto.

Existe também a possibilidade de criar uma lista em branco, personalizá-la de acordo com as necessidades e reutilizá-la como modelo.

Para armazenar documentos, o SharePoint disponibiliza a criação de bibliotecas de documentos. Estas bibliotecas permitem que os utilizadores descrevam melhor o conteúdo de um documento, através da adição de "*metadata*". A "*metadata*" poderá conter informação tal como: o número de páginas do documento, o cliente a que pertence, ou qualquer outro tipo de informação numérica ou alfanumérica. Não menos importante é a operação de pesquisa sobre a "*metadata*" armazenada, que permite encontrar rapidamente um documento [NOEL e SPENCE, 2007].

Ambos os objectos, listas e bibliotecas de documentos, permitem activar o controlo automático de versões. O controlo de versões permite saber quem, quando e o que foi alterado, quer seja um registo de uma lista ou um documento proveniente de uma biblioteca de documento, conforme mostra a Ilustração 12 [NOEL e SPENCE, 2007].

Ilustração 12 – Histórico de versões de um item de uma lista.

No caso das bibliotecas de documentos existe ainda a possibilidade de activar mecanismos de *check-in* e *check-out*.

Relativamente aos formulários de inserção, edição e visualização é importante referir que estes são gerados automaticamente para cada lista, de acordo com as configurações efectuadas sobre os campos da mesma. A ilustração seguinte apresenta um formulário exemplo para as opções de visualização e edição de um item de uma lista.

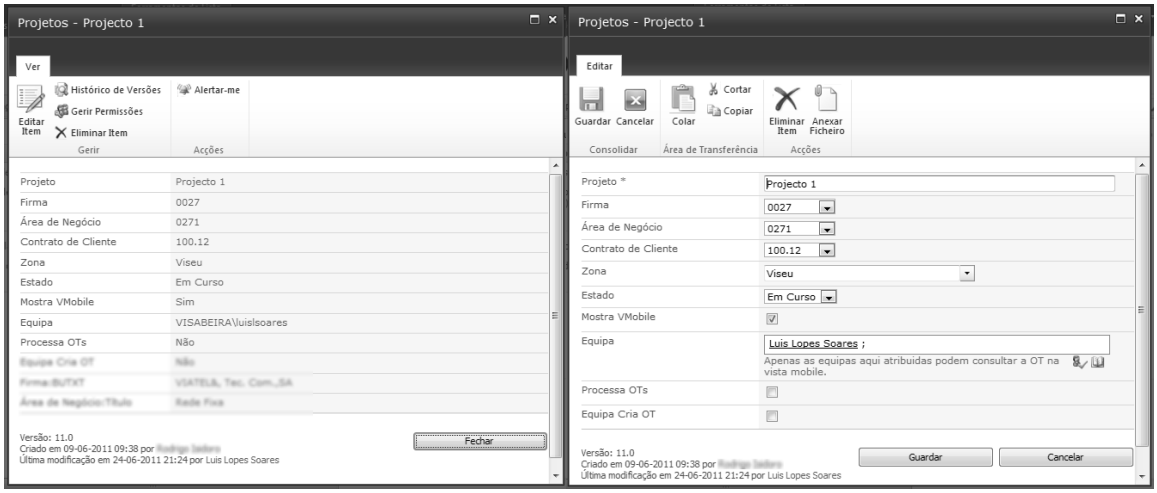

Ilustração 13 – Formulários de visualização e edição de um registo.

A inclusão de funcionalidades desenvolvidas à medida é possível através da adição de *Web Parts*. Conforme é visível na Ilustração 14 existe um número alargado de *Web Parts* que podem ser adicionadas a uma página.

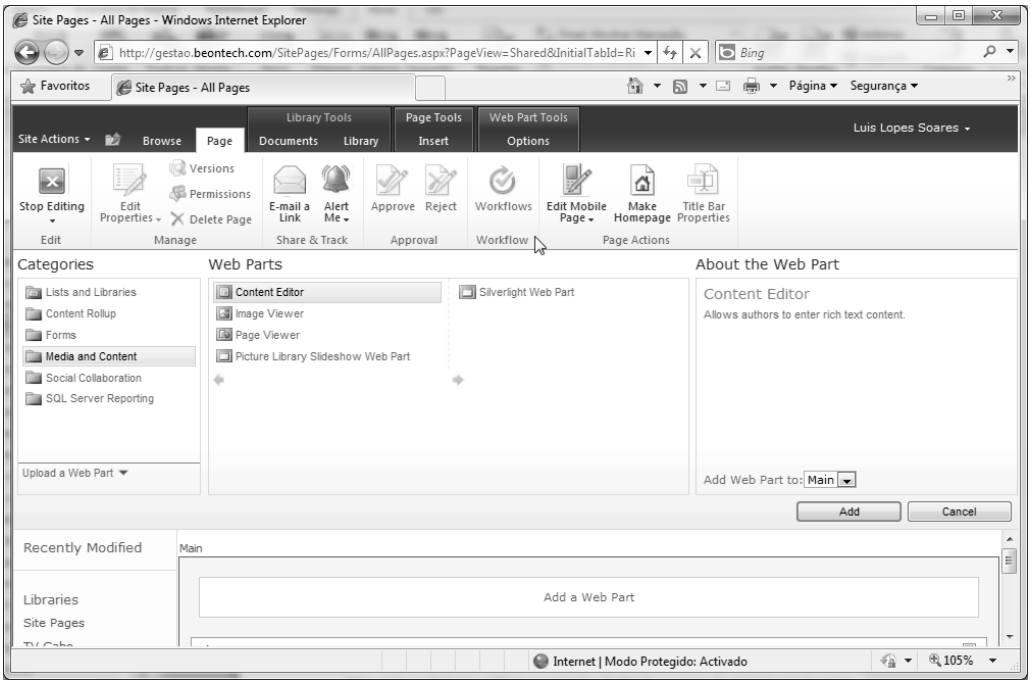

Ilustração 14 – Inclusão de funcionalidade feita à medida através de *Web Parts* do SharePoint.

As *Web Parts* vão desde simples formulários em *HyperText Markup Language* (HTML) a complexas funcionalidade em *Silverlight*. Contudo, estas funcionalidades desenvolvidas à medida e implementadas através de *Web Parts* não funcionam no Microsoft SharePoint Workspace, como iremos perceber mais à frente.

O SharePoint permite ainda a criação de vistas sobre os dados. Uma vista representa um conjunto de colunas que pretendemos ver, sobre as quais podem ser aplicadas restrições e ordenações aos dados que esta apresenta. Estas vistas podem ser públicas ou privadas, limitando assim o acesso às mesmas por terceiros [NOEL e SPENCE, 2007].

Como controlo de acessos, permite a atribuição de níveis de acesso por lista, por utilizador ou por grupos de utilizadores. Os níveis de acesso são completamente personalizáveis em várias categorias, tipo de acções que pode desempenhar (visualizar, inserir, editar, entre outros.), nível de acesso a páginas e conteúdos, etc.

Como meio de acesso a dados, informações de listas, gestão de acessos, administração de sites, entre outros, o SharePoint disponibiliza um conjunto alargado de *Web Services*. Dentro deste conjunto, o *Web Service* "Lists" é aquele que permite interagir e manipular os dados de listas do SharePoint, nomeadamente através de operações de consulta, edição, inserção, entre outras. A ilustração seguinte apresenta alguns dos métodos disponibilizados por este *Web Service*.

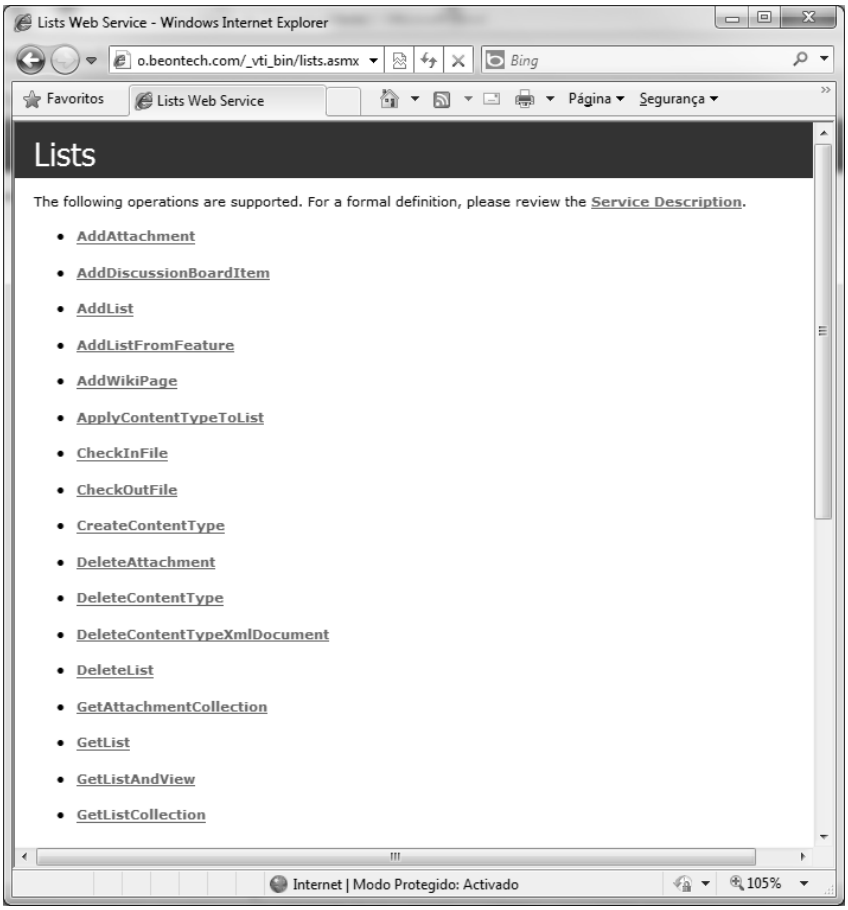

Ilustração 15 – Web Services disponibilizados pelo SharePoint para acesso a listas e seus conteúdos.

Já o Microsoft SharePoint Workspace permite, de forma transparente, a utilização *offline* de dados provenientes de *sites* SharePoint. Ao adicionar um novo *site* ao Microsoft SharePoint Workspace, este abre automaticamente um ecrã de sincronização dos conteúdos do *site*. A Ilustração 16 mostra o estado do processo de sincronização de um *site* SharePoint denominado de Produção.

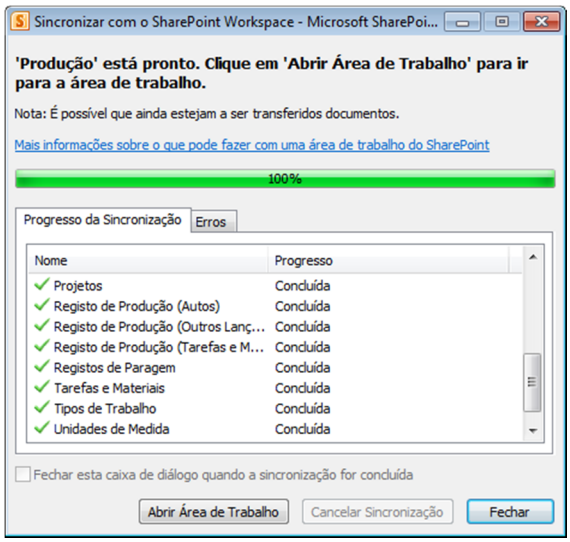

Ilustração 16 – Sincronização de sites SharePoint com o Microsoft SharePoint Workspace.

Durante o processo de sincronização, os dados são descarregados para a máquina do utilizador, permitindo que estes possam depois ser utilizados ou alterados, mesmo sem ligação ao *site* (modo *offline*).

Após sincronização do *site* com sucesso, fica disponível a área de trabalho. Esta área de trabalho é muito simples e encontra-se dividida em quatro zonas distintas, devidamente identificadas na Ilustração 17.

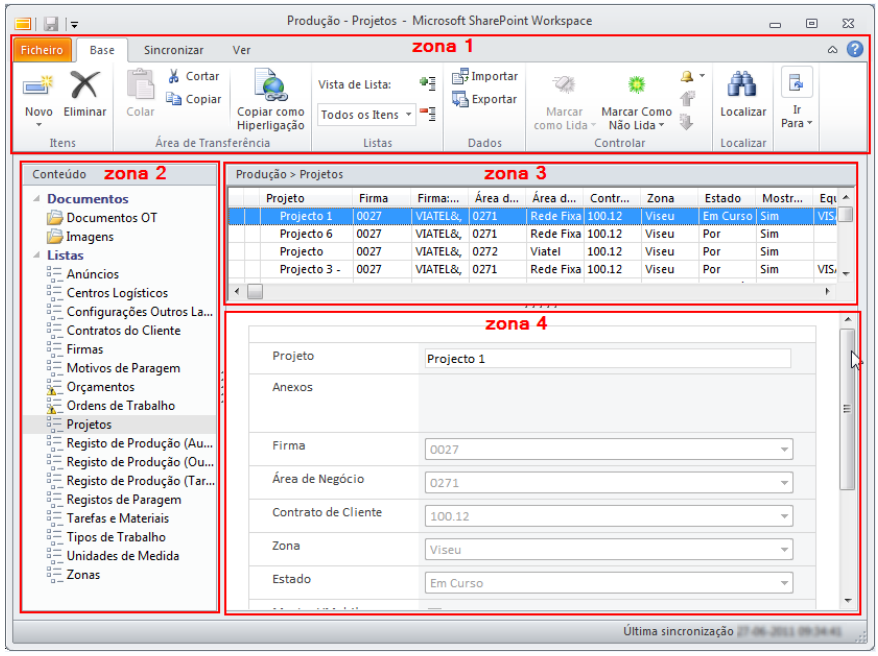

Ilustração 17 – Área de Trabalho do Microsoft SharePoint Workspace.

A zona 1 agrupa um conjunto de opções sobre os dados disponíveis, devidamente agrupadas e sob um friso semelhante ao Microsoft Office na sua versão 2007 ou superior. Na zona 2 são listados todos os conteúdos, que estão ou não sincronizados, do *site* SharePoint. Na Ilustração 17 podemos claramente distinguir dois tipos de conteúdos, bibliotecas de documentos (Documentos OT

e Imagens) e listas de dados. Ao seleccionarmos um dos conteúdos identificados, ficarão disponíveis na zona 3 todos os registos desse conteúdo, e na zona 4 a informação do registo seleccionado na zona 3.

O duplo clique sobre um registo abre a opção de edição do mesmo. Esta opção resulta num formulário automático que implementa as mesmas validações que foram definidas anteriormente na respectiva lista do *site* SharePoint. Na ilustração seguinte podemos facilmente analisar a dinâmica de criação do formulário de acordo com o tipo de dados de cada campo.

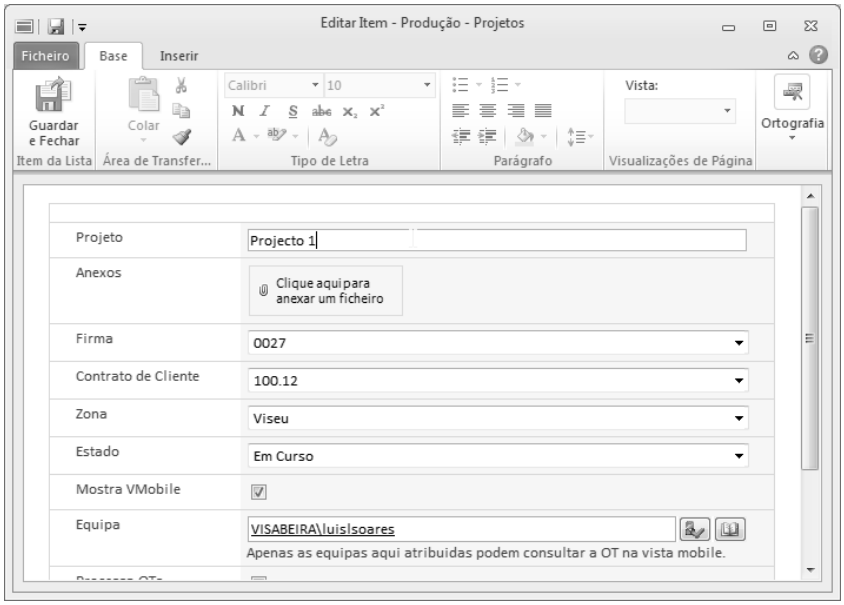

Ilustração 18 – Opção de edição de um registo SharePoint através do SharePoint Workspace.

Todas as alterações sobre os dados serão depois repercutidas, de forma automática ou a pedido do utilizador, no *site* originário e vice-versa.

Após análise sobre esta solução, foram encontradas algumas limitações que inviabilizam o sucesso da sua aplicação neste projecto. A limitação principal, e a mais crítica, advém do desenvolvimento de funcionalidades e dos dados que estas necessitam. O número de ligações entre listas (relacionamentos) é limitado, assim como o desenvolvimento de novas funcionalidades, uma vez que o SharePoint Workspace não será capaz de abrir funcionalidades desenvolvidas à medida. De igual forma, não temos controlo sobre o modo como os dados são guardados localmente. Sem a capacidade de abrir funcionalidades desenvolvidas à medida, e sem controlo sobre os dados armazenados localmente a criação e impressão de Guias de Transportes torna-se impossível. Relativamente ao Sistema Integrador esta solução não apresenta nenhuma ferramenta que seja capaz de desempenhar esse papel, ou seja, não permite estabelecer uma ligação a outro Sistema de Informação e efectuar a respectiva integração de dados.

2.2 SalesForce

A outra solução encontrada tem o nome de SalesForce. Esta é uma empresa bastante conhecida pelo desenvolvimento personalizado de *software* e pelo seu *Customer Relationship Management* (CRM), com o mesmo nome da empresa. Para além do CRM esta empresa disponibiliza uma plataforma que permite o desenvolvimento de *software* à medida, baseado na tecnologia *Cloud Computing* [SalesForce, 2010].

A plataforma, de nome Force.com, permite a criação de aplicações e *websites* de forma personalizada, rápida e fácil. Todas estas soluções são disponibilizadas pela *Cloud*, não necessitando de qualquer *hardware* ou *software* adicional. Outra vantagem da plataforma Force.com é a execução das aplicações em qualquer plataforma ou dispositivo móvel (Portáteis, PDA's, entre outros) [Force.com, 2010].

Analisando a Ilustração 19 verifica-se que esta plataforma está organizada em quatro produtos principais e uma base de dados, são eles:

- Appforce, um produto que permite o rápido desenvolvimento de aplicações com carácter empresarial;
- Siteforce, um produto responsável por permitir o desenvolvimento de *websites* dinâmicos e com uma grande quantidade de informação;
- Vmforce, que tem como objectivo permitir um rápido desenvolvimento de aplicações empresariais na linguagem de programação Java [Java, 2011];
- ISVforce, apresentado como o produto que fornece o caminho mais rápido para disponibilizar aplicações no mercado de *software*;

Ilustração 19 – Produtos Force.com
(Fonte: [Force.com, 2010]).

Acoplado a estes produtos encontra-se a base de dados de nível empresarial que está disponível na *Cloud*. Esta é utilizada nos dias de hoje por mais de cem mil organizações, entres as quais alguns dos maiores bancos do mundo, agências governamentais e profissionais da área de saúde, sendo assim considerada pela SalesForce como a Base de Dados mais confiável e segura do mundo [Database.com, 2010]. Como foi mostrado na ilustração anterior, todas as *Application Programming Interfaces* (APIs) são baseadas em padrões de código livre.

Analisando esta solução no contexto do presente projecto, dos quatro produtos atrás apresentados apenas dois deles se enquadram neste âmbito, são eles o *Appforce* e o *Vmforce*. Ambos permitem o fácil desenvolvimento de funcionalidades através de objectos *drag and drop*. Além disso, a implementação de lógica de negócio é assegurada por um motor de *workflow*, também visual, que

permite a definição de processos, regras, tarefas, e até mesmo a configuração de alertas. Disponibilizam ainda algumas ferramentas que permitem aos utilizadores finais a criação de relatórios e *dashboards* sobre os dados a que estes possuem acesso [Appforce, 2010] [Vmforce, 2010].

O desenvolvimento de *software* pode ser auxiliado pela existência de três estágios (desenvolvimento, testes e produção), possibilitando assim um desenvolvimento de *software* mais seguro, sem comprometer as funcionalidades que se encontram em operação.

Como auxílio ao desenvolvimento de aplicações, a SalesForce disponibiliza um portal de apoio ao programador, denominado de *developerforce* [Developer Force, 2010]. Neste portal podemos facilmente encontrar todas as ferramentas necessárias para iniciar o desenvolvimento de aplicações. As ferramentas disponíveis vão desde *Integrated Development Environments* (IDE's), ferramentas de acesso a dados, ferramentas para integração de funcionalidades na *Cloud*, entre outras. Para além destas ferramentas, o portal serve também de ponto de partilha de conhecimento, contendo alguns exemplos de projectos e o respectivo código fonte.

Após efectuada a análise sobre os dois produtos anteriormente apresentados (o *Appforce* e o *Vmforce*), verificou-se que ambos são soluções *Web* que operam exclusivamente no modo *online*. Para colmatar esta necessidade e disponibilizar soluções capazes de operar quer no modo *online*, quer no modo *offline*, a SalesForce recorre a uma parceria com a Adobe. Assim, a implementação destes dois tipos de soluções é possível recorrendo ao desenvolvimento de aplicações *Rich Internet Applications* (RIA). A tecnologia que permite este desenvolvimento é o Adobe AIR [Adobe for Force, 2010].

A sua arquitectura de implementação é representada segundo o diagrama apresentado na ilustração seguinte.

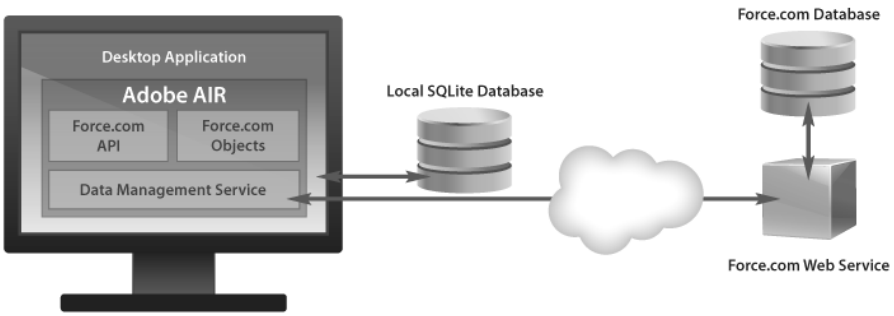

Ilustração 20 – Arquitectura aplicacional de soluções Desktop para Cloud.
(Fonte: [Adobe for Force, 2010])

Como podemos ver na ilustração apresentada, existe um componente *desktop* desenvolvido na tecnologia Adobe AIR e suportado por um conjunto de ferramentas e APIs Force.com.

O desenvolvimento ocorre na linguagem de programação Flex [AdobeFlex, 2011] e é auxiliado com inúmeras ferramentas de produtividade que permitem uma implementação multiplataforma.

Na solução *offline* da SalesForce, os dados são armazenados localmente numa Base de Dados SQLite (Ilustração 20) sendo que estes, são posteriormente sincronizados para a Base de Dados existente na *Cloud* através de ferramentas disponibilizadas pela SalesForce.

Uma vez que esta solução permite o desenvolvimento de funcionalidades à medida, através das linguagens de programação acima identificadas, e dispõe de um módulo de sincronização de dados que permite que a funcionalidade opere no modo *offline*, aparentemente é uma solução que poderia ser adoptada neste projecto. No entanto, existem algumas dúvidas no que diz respeito à atribuição de acessos, actualmente gerida na *active directory* do Grupo Visabeira, e à diversidade das estruturas de dados que cada funcionalidade necessita para operar.

Outra limitação, que coloca em causa a viabilidade desta implementação, encontra-se no facto de não ser tecnologia Microsoft, o que se traduz num custo de licenciamento adicional, quer a nível de *software* para o desenvolvimento de funcionalidades (*Adobe Flash Builder*), quer a nível de contrato com a empresa SalesForce, uma vez que as aplicações e a Base de Dados central terão que ser publicadas e distribuídas na *Cloud*, acrescendo o custo de prestação deste serviço.

A adopção desta solução poderia também pôr em causa o funcionamento do Sistema Integrador central, isto porque, e analisando a Ilustração 21, todas as aplicações clientes estabelecem uma ligação ao Sistema Integrador, sendo que é este mesmo Sistema Integrador que interage com os restantes Sistemas de Informação (GrVisa, SAP e BeOn), através de uma segunda ligação efectuada sobre a Internet. Assim, o funcionamento desta solução depende directamente destas duas ligações.

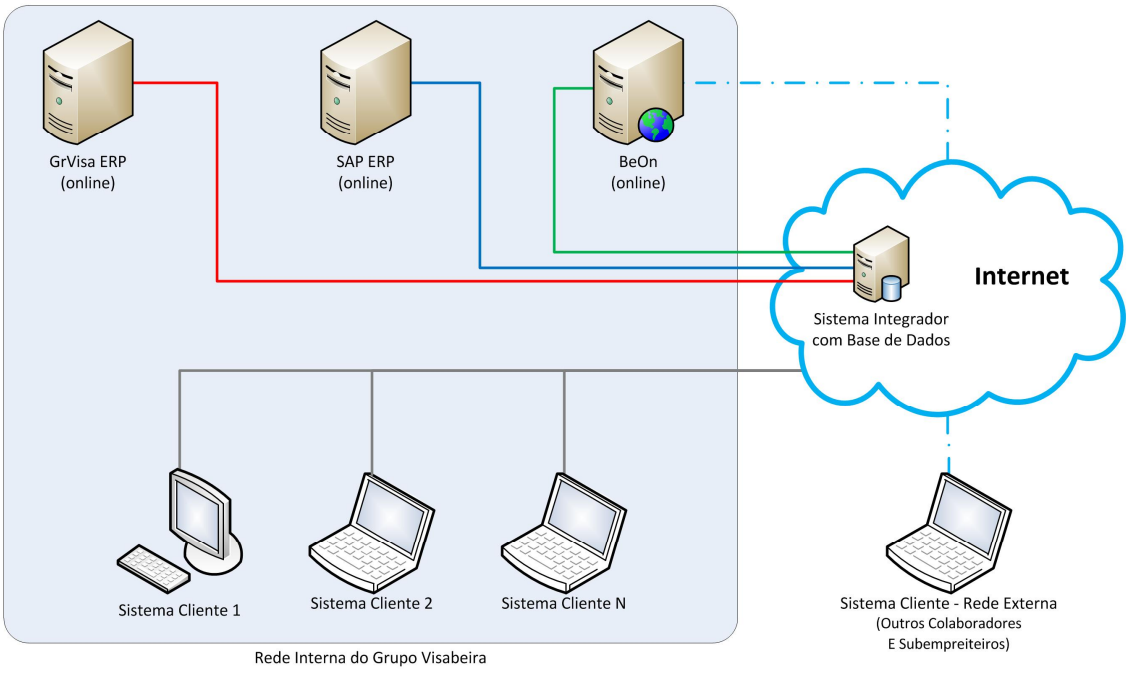

Ilustração 21 – Esquema da arquitectura proposta para a implementação SalesForce.

Como é apresentado na Ilustração 21, e dada a arquitectura projectada para o presente sistema, o Sistema Integrador encontrar-se-ia localizado na *Cloud* e necessitaria de estabelecer duas liga-

ções através da Internet. Uma ligação seria responsável por permitir o acesso das aplicações clientes ao Sistema Integrador. A outra ligação deveria permitir a comunicação entre o Sistema Integrador e o respectivo Sistema de Informação Externo. De uma forma geral, e uma vez que o bom funcionamento desta solução depende directamente da qualidade das duas ligações acima identificadas, conclui-se que a solução poderá não ser muito vantajosa, tendo em conta o número de acessos previsível por parte dos utilizadores, o número de Sistemas de Informação existentes no Grupo Visabeira, o volume de informação gerada diariamente e as limitações atrás identificadas. Outra desvantagem da adaptação desta solução, e talvez a mais crítica, deve-se ao facto de ser uma nova tecnologia, apenas do conhecimento de uma minoria dos programadores da Visabeira Digital. Por outro lado, e uma vez que a linguagem de programação envolvida também não é do conhecimento geral, o desenvolvimento de *software* seria mais demorado e propício a erros.

2.3 Resumo comparativo

O resultado desta pesquisa contribuiu para solidificar conhecimentos, bem como identificar as mais-valias das várias soluções que poderão auxiliar no desenvolvimento deste projecto. Assim, a arquitectura do Microsoft SharePoint será fonte inspiradora na implementação da integração dos módulos de *software* na aplicação informática e na definição do modelo lógico de dados proposto nesta arquitectura, devido ao dinamismo que esta permite.

Na análise efectuada sobre as duas soluções anteriormente apresentadas, verificou-se que ambas apresentam inúmeras características que se enquadram no âmbito deste projecto, nomeadamente na arquitectura cliente/servidor, onde um conjunto de utilizadores (clientes) acedem a um conjunto de dados e aplicações centrais (servidor). O Microsoft SharePoint Workspace possui uma grande dinâmica a gerar os formulários e todas as validações inerentes. De igual forma, permite um funcionamento *offline* e possui um mecanismo de sincronização, capaz de garantir um controlo de versões eficiente e ainda disponibilizar opções de *check-in* e *check-out* sobre documentos. Por outro lado, é muito rígido no desenvolvimento de funcionalidades, este limita-se a gerar formulários com base na estrutura definida nas listas existentes no SharePoint. Não permite a criação de *reports* personalizados para impressão e, como principal limitação, não poderia funcionar como integrador de sistemas de informação, uma vez que não tem qualquer tipo de ferramenta que apoie ou permita a integração com Sistemas de Informação Externos.

Já a solução SalesForce permite um desenvolvimento personalizado das aplicações clientes e garante o funcionamento *offline* através de mecanismos de sincronização. Contudo apresenta um custo de aquisição e manutenção da solução, não só por não ser tecnologia Microsoft, para a qual a Visabeira Digital tem licenciamento, mas também devido à prestação de serviço de alojamento, uma vez que a aplicação será distribuída na *Cloud* e não nos servidores internos do Grupo Visabeira. O funcionamento do Sistema Integrador teria de ser implementado como funcionalidade, na qual seriam parametrizadas as várias ligações aos servidores do Grupo e agendadas as respectivas rotinas de integração. Contudo, esta solução não é, de todo, uma solução robusta podendo pôr em causa o seu bom funcionamento, principalmente devido a falhas de ligação ou comunicação entre a *Cloud* e os servidores do Grupo Visabeira. Tendo em conta o número de acessos previsível, por parte dos utilizadores, o número de Sistemas de Informação existentes no Grupo Visabeira e o

volume de informação gerada diariamente, podemos concluir que este sistema é crítico para o Grupo Visabeira, uma vez que se este ficar indisponível, uma grande quantidade de informação e de utilizadores serão imediatamente afectados. Por outro lado, o facto de alguma informação ficar distribuída na *Cloud* não inspira grande confiança por parte das empresas do Grupo.

Com esta análise, e de acordo com a especificidade do problema e dos objectivos da empresa, o desenvolvimento interno do sistema proposto é conveniente.

Capítulo 3

Estudo de uma nova arquitectura

O capítulo 3 apresenta detalhadamente o estudo efectuado para a proposta da nova arquitectura. Este introduz os sistemas envolvidos, as principais características, as interfaces adoptadas, bem como os canais de comunicação estabelecidos entre os diversos sistemas.

3.1 Sistemas envolvidos

Da discussão atrás tecida, nomeadamente quando foram formulados os objectivos do sistema, e realizado um esforço de idealização e concepção, criou-se uma possível arquitectura cujo diagrama e respectivos componentes são apresentados na Ilustração 22.

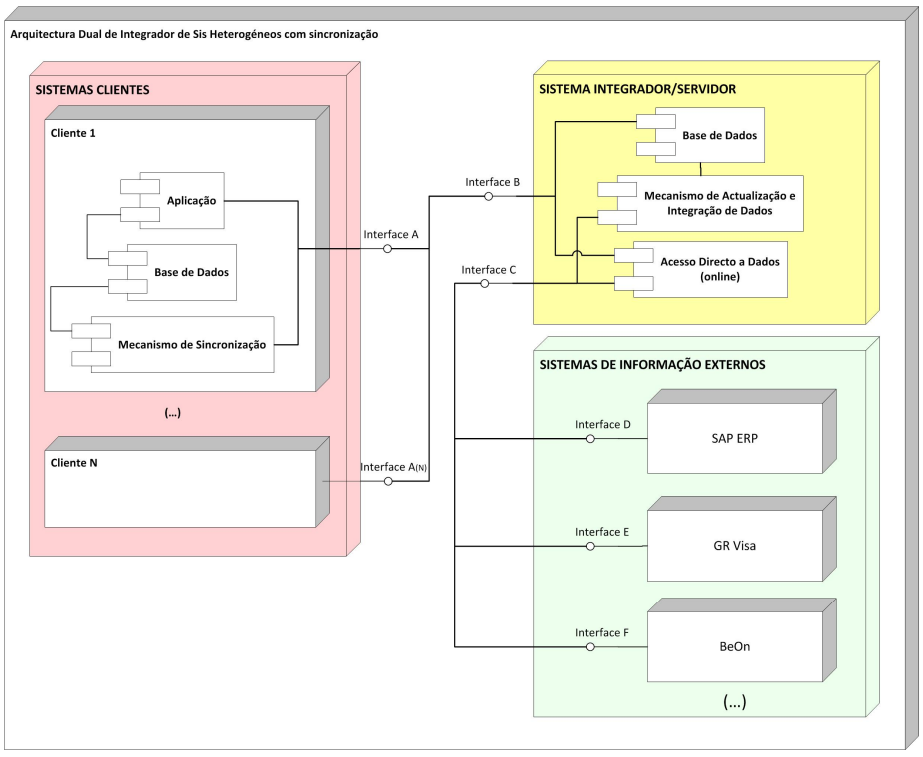

Ilustração 22 – Diagrama geral dos componentes da arquitectura proposta.

Como se pode ver na ilustração acima apresentada, a arquitectura proposta encontra-se dividida em três sistemas: Sistema Cliente, Sistema Integrador/Servidor e, por fim, os Sistemas de Informação Externos. O Sistema Cliente é constituído por uma Aplicação, uma Base de Dados e um Mecanismo de Sincronização de dados. Já o Sistema Integrador/Servidor é constituído por uma Base de Dados, um Mecanismo de Actualização e Integração de Dados, e ainda um componente responsável pelo Acesso Directo a Dados de outros Sistemas de Informação. Finalmente, os Sistemas de Informação Externos representam todos os Sistemas de Informação do Grupo Visabeira, com os quais o Sistema Integrador/Servidor comunica para obter/integrar informação. Todos os componentes constituintes dos sistemas acima apresentados serão abordados durante as próximas subsecções.

3.1.1 Sistema Cliente

O Sistema Cliente representa um dispositivo terminal a que os utilizadores recorrem para efectuar operações. No caso particular que desencadeou este projecto, o exemplo de uma operação do Sistema Cliente é a emissão de Guias de Transporte. Como se depreende da arquitectura, será possível que um outro qualquer subsistema possa vir a ser incluído sob o formato de módulo de *software*. Como, por exemplo, o subsistema de validação da recepção de Guias de Transporte pelos armazéns.

Este sistema possuirá uma Aplicação, uma Base de Dados e um Mecanismo de Sincronização interligados conforme a Ilustração 23.

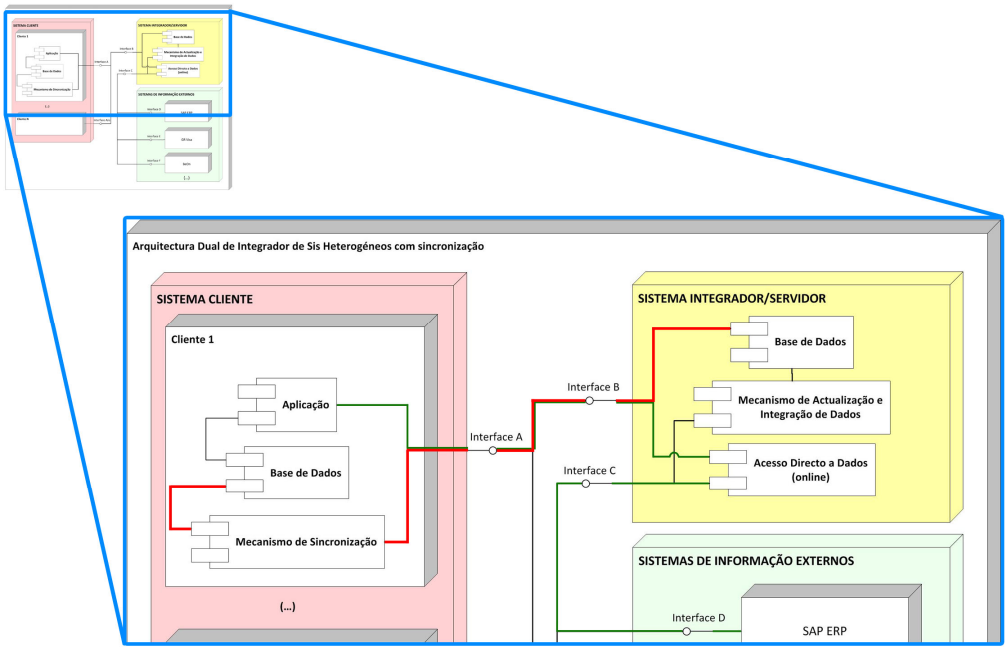

Ilustração 23 – Mecanismo de sincronização do Sistema Cliente com o Sistema Servidor.

A aplicação será responsável apenas por apresentar as funcionalidades a que cada utilizador tem acesso. Disponibilizará ainda um conjunto de operações que permitirão invocar, de forma manual, o mecanismo de sincronização de dados entre o Sistema Cliente e o Sistema Integrador/Servidor.

A Base de Dados terá a seu cargo o armazenamento dos dados e das funcionalidades a disponibilizar na aplicação. Pretende-se que o sistema seja capaz de armazenar dados que podem ter estruturas muito diferentes de funcionalidade para funcionalidade. De igual forma, deverá garantir a disponibilidade de todos os dados necessários para a execução de funcionalidades no modo *offline*, i.e., na situação de inexistência de uma ligação ao sistema central.

Por fim, o Mecanismo de Sincronização terá a responsabilidade de sincronizar com o Sistema Integrador/Servidor, periodicamente ou a pedido do utilizador, os novos registos e os dados de apoio às funcionalidades que poderão operar no modo *offline*. Este mecanismo está visível na Ilustração 23, onde as linhas de cor vermelha representam a sua comunicação com a Base de Dados do Sistema Integrador/Servidor, através da ligação entre as interfaces A e B.

No modo *online*, o sistema deverá permitir consultas e efectuar operações, em tempo real, sobre os dados dos diversos Sistemas de Informação Externos. Para atingir tal objectivo, a aplicação estabelece uma comunicação com o componente "Acesso Directo a Dados" do Sistema Integrador/Servidor, através das mesmas interfaces A e B. Após estabelecer a comunicação, a aplicação indica os dados que pretende obter do Sistema de Informação Externo. Por sua vez, o Sistema Integrador/Servidor determina qual o Sistema de Informação Externo e qual o modo para consultar/efectuar a operação solicitada. Através da interface C, este envia o pedido de informação para o respectivo Sistema de Informação Externo que devolve a informação referente à operação solicitada. A informação é depois encaminhada para o Sistema Cliente, através da interface B. Após recepção da informação por parte do Sistema Cliente, a funcionalidade já pode proceder ao seu tratamento e apresentação ao utilizador final. Todo este fluxo de dados é representado na Ilustração 23 em linhas de cor verde.

São estes dois tipos de acesso a dados (*offline* – recorrendo ao processamento e ao acesso a dados locais actualizados pelo mecanismo de sincronização e *online* – através do Sistema Integrador/Servidor) que caracterizam o sistema dual da arquitectura proposta. Assim é possível implementar funcionalidades que acedem a dados no modo *online*, e aquando falhas de ligação passam a operar no modo *offline*, registando a informação na Base de Dados local para posteriormente ser sincronizada e integrada. Desta forma conseguimos ter um sistema robusto que garanta o seu funcionamento mesmo em situações de falha de ligação com os Sistemas de Informação principais.

3.1.2 Sistema Integrador/Servidor

O segundo sistema, Sistema Integrador/Servidor é composto por uma Base de Dados, um Mecanismo de Actualização e Integração de Dados e um componente de Acesso Directo a Dados, como apresenta a Ilustração 22.

A Base de Dados armazenará a informação de apoio que estará disponível para os Sistemas Clientes sincronizarem e utilizarem no modo *offline*. Para além dos tradicionais dados de apoio (Firmas, Centros Logísticos, Materiais, etc.), utilizados no registo de informação em modo *offline*, este disponibilizará também as diversas funcionalidades e os acessos dos vários utilizadores. Por exemplo, no caso da emissão de uma Guia de Transporte em modo *offline*, uma das tabelas de apoio necessária é a tabela de materiais. Uma vez que a gestão de materiais é feita em SAP no

módulo MM, o Sistema Integrador/Servidor é responsável por obter essa informação do Sistema SAP (através do Mecanismo de Actualização de Dados) e armazená-la na sua Base de Dados local, para posteriormente poder ser sincronizada com os Sistemas Clientes. O mesmo processo acontece no caso dos acessos dos utilizadores e das próprias funcionalidades, uma vez que a sua gestão também será feita num Sistema de Informação Externo (BeOn). Por outro lado, quando um Sistema Cliente entra no modo *offline*, todos os registos criados por este são armazenados na Base de Dados local ao Sistema Cliente. Posteriormente, numa primeira fase, quando o Sistema Cliente entra no modo *online*, os dados são sincronizados com a Base de Dados do Sistema Integrador/Servidor. Numa segunda fase, todos estes registos serão, em modo assíncrono, integrados com o respectivo Sistema de Informação Externo. A principal vantagem deste processo é o facto de permitir que os Sistemas Clientes estejam *online* apenas durante um curto período de tempo (o necessário para sincronizar os dados com o Sistema Integrador/Servidor e não o necessário para garantir todo o processo de integração com o respectivo Sistema de Informação Externo). Caso o processo de integração falhe, a informação desta ocorrência é armazenada na Base de Dados e o registo assinalado como contendo erro, para que mais tarde possa ser corrigido e reintegrado.

Neste sistema, o Mecanismo de Actualização e Integração de Dados será o responsável por garantir a integração dos dados provenientes de Sistemas Clientes com os diversos Sistemas de Informação Externos, garantindo a sua integridade e o registo das operações efectuadas. Será o único sistema a comunicar e a interagir directamente com os Sistemas de Informação Externos. Este mecanismo é também responsável por manter os dados de apoio, disponíveis na Base de Dados, devidamente actualizados.

Ao analisarmos a Ilustração 24 facilmente percebemos que o Mecanismo de Actualização e Integração de Dados funciona nos dois sentidos. Isto é, interliga-se aos Sistemas de Informação Externos através da interface C com o intuito de proceder à actualização dos dados de apoio existentes na Base de Dados do Sistema Integrador/Servidor. Por outro lado, este mesmo mecanismo consulta os registos, provenientes do Sistema Cliente e armazenados na Base de Dados, e integra-os com os respectivos Sistemas de Informação Externos (GrVisa, SAP ERP, BeOn), actualizando o seu estado de acordo com o sucesso ou não da operação. Neste processo de integração, o mecanismo será capaz de interpretar cada registo e gerar a respectiva transacção no Sistema de Informação de destino. Ambos os fluxos são visíveis na Ilustração 24 através das linhas de cor vermelha.

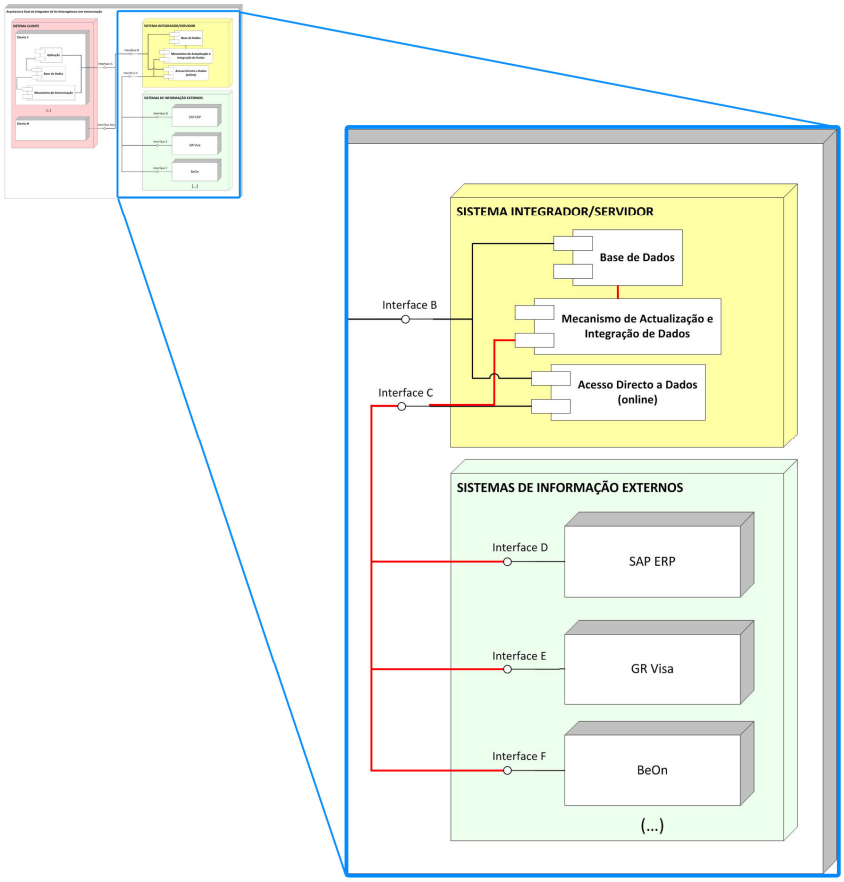

Ilustração 24 – Fluxo de actualização e integração de dados entre o Sistema Servidor e os SI Externos.

Em ambos os casos (actualização ou integração de dados), o Mecanismo de Actualização e Integração de Dados opera sob um conjunto de parametrizações e configurações que permitem definir a periodicidade das várias operações.

O componente de Acesso Directo a Dados encontra-se devidamente abordado na subsecção anterior (3.1.1).

3.1.3 Sistemas de Informação Externos

Por fim, e não menos importante, os Sistemas de Informação Externos. Estes são sistemas tipicamente constituídos por Bases de Dados e *software* aplicacional. Os Sistemas de Informação Externos são extremamente robustos e complexos. Neste projecto são a fonte de onde serão extraídos os dados de apoio para utilização pelas funcionalidades no modo *offline*, e o ponto de destino do Sistema Integrador/Servidor, no processo de integração de dados. Esta integração é efectuada pela interface C, visível na Ilustração 24, sobre a qual é feito o mapeamento para o respectivo Sistema de informação Externo.

A arquitectura proposta neste projecto permite a comunicação entre todos os Sistemas de Informação Externos de forma centralizada e mediada pelo Sistema Integrador/Servidor. Esta centralização permite eliminar situações de replicação do código de acesso a dados, integração e lógica de negócio implementada actualmente nas diversas aplicações clientes. O processo demorado de

manutenção e replicação do código pelas diversas funcionalidades com comportamentos semelhantes, atrás referido, será extinto e assegurado unicamente pelo Sistema Integrador/Servidor, uma vez que toda a comunicação e informação transferida entre os Sistemas de Informação Externos e os Sistemas Clientes passa por ele.

No âmbito deste projecto, os principais Sistemas de Informação Externos identificados são o GrVisa, o SAP ERP e o BeOn.

3.2 Principais características

Como principais características desta arquitectura identificam-se as seguintes:

- Sistema dual, uma vez que o Sistema Cliente é capaz de funcionar em modo *online* e *offline*. No modo *online* permite aceder, em tempo real, a dados armazenados nos respectivos Sistemas de Informação. Já no modo *offline*, o sistema garante a operacionalidade de funcionalidades *offline*, isto é, funcionalidades capazes de operar exclusivamente na máquina cliente. Estas funcionalidades recorrem à Base de Dados local para validar os acessos dos utilizadores e operar sobre os dados necessários ao seu funcionamento;
- Permite a sincronização de dados de forma automática ou a pedido do utilizador;
- Integração de forma automática entre os dados provenientes do Sistema Cliente e os diversos Sistemas de Informação Externos;
- Centralização da lógica de negócio num único nó, o Sistema Integrador/Servidor;
- Desenvolvimento de *software* de forma modular e isolada, através da integração de funcionalidades sob a forma de pequenos módulos de software;
- Possibilidade de expansão, não só do número de utilizadores do Sistema Cliente, mas também de novos Sistemas de Informação Externos e novos clientes heterogéneos (v.g. dispositivos móveis), uma vez que o Sistema Integrador/Servidor irá garantir a integridade dos dados entre todos os Sistemas de Informação envolvidos.

3.3 Interfaces e Canais de comunicação

A principal característica de uma interface é permitir a interligação de sistemas. Neste caso, podemos claramente identificar três interfaces principais, visíveis na Ilustração 22. As interfaces A e B, entre o Sistema Cliente e o Sistema Integrador/Servidor, e a interface C, entre o Sistema Integrador/Servidor e os Sistemas de Informação Externos. E, ainda que de uma forma não tão explícita, pode supor-se uma interface existente no Sistema Cliente, entre a Base de Dados e a aplicação / mecanismo de sincronização.

As interfaces A e B são as responsáveis por permitir a troca de informação (dados e funcionalidades) entre os Sistemas Clientes e o Sistema Integrador/Servidor.

Por outro lado, e uma vez que se pretende que o único sistema a comunicar e a interagir com os diversos Sistemas de Informação Externos seja o Sistema Integrador/Servidor, a interface C

deverá ter um carácter adaptativo. Esta interface deverá ser capaz de actuar com o respectivo método de obtenção ou integração de dados consoante o Sistema de Informação Externo.

Relativamente aos canais de comunicação, esta arquitectura pretende distinguir e classificar dois canais com quatro funções distintas. O primeiro canal é estabelecido entre o Sistema Cliente e o Sistema Servidor, claramente identificado na Ilustração 23. Este possui duas funções, uma de actualização dos dados de apoio disponíveis no Sistema Cliente e outra de envio dos novos registos do Sistema Cliente para o Sistema Integrador/Servidor, que ficam transitoriamente armazenados na Base de Dados do Sistema Integrador/Servidor.

O segundo canal de comunicação estabelece-se entre o Sistema Integrador/Servidor e os Sistemas de Informação Externos, onde a sua existência tem como objectivo a actualização dos dados de apoio, provenientes dos Sistemas de Informação Externos e, que serão disponibilizados aos Sistemas Clientes. Outro objectivo deste canal de comunicação é a integração dos dados provenientes dos Sistemas Clientes, previamente armazenados em Base de Dados, com os respectivos Sistemas de Informação Externos. O fluxo de informação anteriormente descrito é representado na Ilustração 24.

No capítulo seguinte serão apresentados mais detalhadamente as interfaces e os canais de comunicação implementados na arquitectura proposta para este projecto.

3.4 Armazenamento dos dados

No que diz respeito ao armazenamento de dados, o sistema pode ser dividido em duas partes. A primeira refere-se aos dados relativos ao funcionamento base da aplicação, nomeadamente funcionalidades, acessos e utilizadores. Esta tem uma estrutura mais rígida e constante. Já a segunda parte tem um carácter mais diverso e dinâmico, isto porque se refere à estrutura dos dados que cada funcionalidade irá gerar. Assim sendo, o sistema deverá ser capaz de armazenar as mais diversas estruturas de dados de forma transparente. De igual modo, as diferentes estruturas de dados geradas no Sistema Cliente deverão poder ser sincronizadas com o Sistema Integrador/Servidor e devidamente integradas com o(s) respectivo(s) Sistema(s) de Informação Externo(s). Este carácter dinâmico é também de extrema importância no armazenamento dos dados de apoio que cada funcionalidade necessita para operar. Por exemplo, a tabela dos armazéns disponíveis contém uma estrutura de campos completamente diferente da tabela de materiais, necessária para a emissão de Guias de Transporte. Este assunto será retomado no capítulo seguinte, referente à implementação da nova arquitectura.

Capítulo 4

Implementação da nova arquitectura

O capítulo agora iniciado visa dar a conhecer a fase de implementação da nova arquitectura. Entre outros assuntos, mostra-se a análise comparativa efectuada entre as várias tecnologias e ferramentas adoptadas. É também apresentada a aplicação, fruto da implementação da nova arquitectura. E, por fim, enumeram-se algumas dificuldades que surgiram durante o processo de implementação desta solução.

4.1 Tecnologias e ferramentas disponíveis

Relembrando o que foi dito atrás, o estudo da arquitectura proposta neste projecto está dividido em três sistemas principais: o Sistema Cliente, o Sistema Integrador/Servidor e os Sistemas de Informação Externos. Desta forma, pretende-se nesta secção identificar as várias tecnologias e ferramentas que poderão ser utilizadas nos sistemas acima identificados.

A análise comparativa de tecnologias e ferramentas disponíveis incide sobre três áreas principais. O desenvolvimento das aplicações informáticas (presente no Sistema Cliente), o Sistema Gestor de Base de Dados (presente no Sistema Cliente e no Sistema Integrador/Servidor), no qual a informação será processada e armazenada, e ainda, as tecnologias disponíveis para estabelecer a comunicação e transferência de dados entre os diversos sistemas da arquitectura. Relativamente ao desenvolvimento do Sistema Integrador/Servidor, este será abordado mais à frente, na secção 4.4. Para cada área serão tidos em conta os requisitos funcionais e não funcionais do sistema a implementar.

4.1.1 Desenvolvimento de aplicações informáticas

A aplicação informática desempenha um papel muito importante na aceitação deste projecto, uma vez que é esta que está em contacto com os utilizadores e reflectirá directamente o agrado e satisfação ou o descontentamento dos mesmos [NIELSEN, 1993].

Importa portanto identificar tecnologias e ferramentas capazes de transmitir a maior satisfação aos utilizadores. Esta satisfação pode ser atingida se as funcionalidades seguirem normas e *guidelines* de usabilidade, quer a nível gráfico quer a nível funcional [BARFIELD, 1993].

A primeira questão surge com o tipo de aplicação que se pretende implementar, isto é, uma aplicação *desktop* ou um consenso entre aplicação *desktop* e aplicação *Web*, mais conhecido por RIA [AdobeRIA, 2011]. Na vertente *desktop* destacamos duas tecnologias: Java e .NET (Windows Forms [WindowsForms, 2011] ou WPF [WPF, 2011]).

A tecnologia Java é conhecida pelas suas aplicações robustas mesmo em máquinas com características reduzidas. A sua versatilidade, eficiência, portabilidade de plataforma e segurança fazem do Java a tecnologia ideal para a computação em rede [HUGHES et. al, 1997].

Já a tecnologia .NET, embora dependente de sistemas operativos Microsoft e da respectiva *framework*, é muito conhecida pela facilidade de programação. As classes Windows Forms existentes na *framework* .NET foram desenhadas para o desenvolvimento de interfaces gráficas denominadas de *Graphical User Interfaces* (GUI). Facilmente se desenvolve uma aplicação recorrendo a janelas, botões, menus, barras de ferramentas e outros elementos visuais disponibilizados nestas classes. A tecnologia *Windows Presentation Foundation* (WPF) fornece um modelo de programação único, que permite a construção de aplicações para o Windows de elevada qualidade gráfica, com recurso a documentos e conteúdos multimédia [WPF, 2011]. Com o WPF é possível desenvolver aplicações autónomas (*standalone*) ou mesmo aplicações capazes de funcionar de modo incorporado num *browser*. Como características adicionais identificamos a possibilidade de utilizar aceleração gráfica por *hardware*, visualização interactiva de dados, e suporte a características do sistema operativo Windows 7, tais como o suporte multi-toque [WPF, 2011]. Para definir a interface recorre à linguagem *Extensible Application Markup Language* (XAML) que permite separar a definição gráfica da interface da lógica de execução da aplicação, recorrendo a ficheiros de código separados (*code-behind files*). O XAML é definido por objectos, atributos, propriedades, eventos e valores, segundo uma estrutura semelhante à do *Extensible Markup Language* (XML) [XAML, 2011].

Na vertente RIA, as tecnologias que mais se destacaram foram: Java FX [JavaFX, 2011], Flex e .NET (WPF e Silverlight [Silverlight, 2011]). As tecnologias JavaFX e Flex são multiplataforma, o que se traduz numa vantagem em relação a tecnologias .NET devido ao aparecimento de novos Sistemas Operativos gratuitos. Contudo, poderão ocorrer algumas incompatibilidades quando a aplicação é utilizada através de um *browser*. Os *browsers* não interpretam a mesma informação do mesmo modo, o que torna o desenvolvimento para a *Web* um pouco mais cuidadoso, com o objectivo de garantir a máxima compatibilidade entre os *browsers* das diversas marcas [MUSCIANO e KENNEDY, 2000].

Em síntese, a principal vantagem da vertente RIA em relação à vertente *desktop*, prende-se com o facto de poder usufruir da mesma aplicação também através de um *browser*, mesmo quando o computador de acesso não é o habitual.

Analisadas as diversas tecnologias abordadas anteriormente, não houve nenhuma que se destacasse das restantes. Tendo isto em conta, e uma vez que a Visabeira Digital é uma empresa certificada pela Microsoft, onde todo o seu desenvolvimento de *software* passa pela utilização de tecnologias Microsoft, esta impôs como limitação a utilização exclusiva destas tecnologias. A sua decisão surgiu pelo facto de, como é dito atrás, nenhuma das tecnologias avaliadas (excluindo as tecnologias Microsoft) se ter destacado claramente.

Dito isto, não foi muito difícil seleccionar a tecnologia a adoptar para a implementação da vertente aplicacional deste projecto. Como tal, adoptou-se a tecnologia Windows Forms, pela sua simplicidade e pela existência de um maior número de colaboradores da Visabeira Digital conhecedores desta tecnologia. É de relembrar que um dos objectivos passa pela migração de funcionalidades existentes (verticais e sem capacidade de integração com SAP) para a nova arquitectura. Deste modo, torna-se indispensável que a maioria dos colaboradores esteja familiarizada com a nova tecnologia, minimizando o tempo necessário para o desenvolvimento das funcionalidades.

Por outro lado, uma vez que um dos requisitos do projecto é a expansão e implementação de funcionalidades através de módulos de *software*, optou-se pela implementação de ficheiros DLL na tecnologia .NET. Estes ficheiros são facilmente criados através de um projecto *Class Library* disponível no Microsoft Visual Studio.

Com o intuito de melhorar o aspecto visual da interface gráfica das aplicações, efectuou-se uma análise global sobre as ferramentas e os componentes de produtividade mais utilizados no desenvolvimento de aplicações. Destacaram-se duas: Telerik [Telerik, 2011] e Developer Express [DevExpress, 2011], ambas de características idênticas. Contudo as ferramentas da empresa Developer Express demonstram ser a melhor aposta, não só pelo enorme número de utilizadores, mas também pela base de conhecimento partilhada e pelo suporte prestado pela equipa de desenvolvimento. Aliada a esta mais-valia, está a enorme diversidade de controlos disponíveis para as diversas tecnologias (Windows Forms, ASP.NET, WPF, Silverlight e Delphi) e ainda, um conjunto de ferramentas (*add-ins*) para o IDE Visual Studio [VisualStudio, 2011], utilizado no desenvolvimento de *software* pela Visabeira Digital [DevExpress, 2011].

Uma vez que a interface gráfica de uma aplicação possui um poder determinante na fase da sua aceitação, foram escolhidos os componentes da Developer Express. Esta escolha advém não só do aspecto gráfico que os componentes demonstram, mas também pelo facto da Visabeira Digital já possuir um licenciamento anual destes componentes, e demonstrar um grande agrado na utilização dos mesmos. Por outro lado, as aplicações actuais também utilizam estes componentes pelos quais os utilizadores sempre demonstraram grande satisfação na sua utilização.

4.1.2 SGBD (Sistemas Gestores de Bases de Dados)

A escolha do Sistema Gestor de Base de Dados teve em atenção dois dos sistemas presentes na arquitectura proposta, o Sistema Cliente e o Sistema Integrador/Servidor. Esta é sem dúvida uma escolha de grande importância, não tão decisiva para o Sistema Cliente quanto para o Sistema Integrador/Servidor. No estudo sobre as soluções disponíveis na actualidade, tivemos em conta as seguintes necessidades:

- Sistemas Gestores de Bases de Dados compactos, que permitam o funcionamento em máquina com desempenhos mais limitados, enquadrados no Sistema Cliente;
- Sistemas Gestores de Bases de Dados com elevado desempenho, capazes de suportar Bases de Dados de elevada dimensão e um volume significativo de pedidos de acesso. Este SGBD será instalado no Sistema Integrador/Servidor.

Relativamente ao primeiro ponto, o SGBD deve responder a alguns requisitos tais como a facilidade de instalação, minimizar o consumo dos recursos da máquina portadora e ainda permitir a sua administração remota, caso seja necessário efectuar uma intervenção sobre o mesmo.

Após uma breve análise sobre as várias soluções do mercado, houve três soluções que se destacaram, entre as concorrentes. As soluções apresentadas são gratuitas e de características semelhantes, são elas: o Oracle 10g Express Edition [Oracle10gXE, 2011], o Microsoft SQL Server Compact [MSSQLServerCompact, 2011] e o Microsoft SQL Server 2008 R2 Express [MSSQLServerExpress, 2011].

Destes três SGBD, o Oracle é conhecido pela sua robustez, elevado desempenho e diversidade de tipos de dados. As principais limitações identificadas são o facto de permitir uma única base de dados por máquina, 4GB para o armazenamento de dados e recurso a apenas um CPU, mesmo em máquinas multi-processador.

Já o Microsoft SQL Server Compact destaca-se pela portabilidade na sua instalação. Este é composto por um ficheiro de tamanho reduzido. Suporta instalações por ClickOnce, MSI, CAB, entre outros sistemas incorporados sem opções avançadas de instalação. Funciona sobre todas as plataformas Microsoft Windows. Como limitação, identifica-se o facto de não disponibilizar ferramentas de administração local e escassas opções no caso de administração remota [MSSQLServerCompact, 2011].

A solução mais completa, das três identificadas, é o Microsoft SQL Server 2008 R2 Express. Este, embora necessite um pouco mais de recursos, possui uma enorme capacidade de processamento e conta com um limite de 10GB de espaço para o armazenamento de dados. Em relação à versão Compact, a versão Express dá suporte a um conjunto rico em tipos de dados SQL Server. Permite também uma instalação rápida, transparente e personalizada através da execução de *scripts*.

Após a análise das três soluções acima identificadas, chegou-se à conclusão que o SGBD mais adequado seria o Microsoft SQL Server 2008 R2 na sua versão Express. Este, para além de ser um produto Microsoft, atinge todos os requisitos identificados para o SGBD do Sistema Cliente.

Relativamente ao segundo ponto, Sistemas Gestores de Bases de Dados com elevado desempenho, as soluções não diferem muito das anteriores. Nas respectivas versões Enterprise destacam-se o Oracle Database 11g Release 2 [Oracle11gR2, 2011] e o Microsoft SQL Server 2008 R2 Enterprise [MSSQLServerEnterprise, 2011]. Ambas possuem uma gestão avançada de segurança, novas melhorias não só na optimização e no armazenamento de dados, mas também no desempenho e escalabilidade do sistema, permitindo a criação de *clusters* capazes de responder a qualquer

falha do sistema. Sem dúvida que qualquer uma das soluções abordadas será uma boa aposta para implementar no Sistema Integrador/Servidor.

Em síntese, pretende-se que o SGBD a implementar no Sistema Integrador/Servidor tenha características que permitam aumentar e potencializar o seu desempenho, garantir configurações avançadas de segurança, permitir a escalabilidade do sistema e optimizar o armazenamento dos dados (no tempo de acesso e no espaço ocupado). A escolha deste SGBD foi directamente influenciada pela Visabeira Digital, uma vez que esta já era detentora de alguns licenciamentos, nomeadamente do SGBD Microsoft SQL Server 2008 R2 Standard, recaindo a escolha, mais uma vez, num produto Microsoft.

4.1.3 Comunicação e transferência de dados

No assunto desta subsecção, pretende-se apresentar um conjunto de tecnologias e ferramentas que permitam estabelecer uma ligação e transferir os dados, de forma automática ou a pedido, entre Sistemas de Informação. A análise efectuada a soluções sobre este assunto foi dividida em dois contextos:

- Tecnologias e ferramentas capazes de estabelecer um canal de comunicação entre Sistemas de Informação;
- Tecnologias e ferramentas que permitam a transferência e sincronização de dados entre diversos Sistemas de Informação.

No contexto de estabelecer um canal de comunicação entre diversas aplicações de *software*, as atenções recaem sobre os *Web Services* [WebServices, 2011] e os *Windows Communication Foundation Services* (WCF Services) [WCFServices, 2011].

Os *Web Services* permitem que o acesso das aplicações, através de protocolos *Web* omnipresentes e formato de dados como o *HyperText Transfer Protocol* (HTTP), o *Simple Object Access Protocol* (SOAP) e o XML, sejam efectuados de forma simples e sem a preocupação da linguagem de programação e da estrutura em que o *Web Service* está implementado [WebServices, 2011].

No modelo de programação .NET os *Web Services* são conhecidos como *ASP.NET Web Services* (ASMX). Estes permitem uma fácil utilização multi-plataforma a nível de linguagem de programação, sistema operativo e diversidade de aplicações informáticas [MANES, 2003].

Já os *WCF Services* aparecem como sendo a última geração de plataformas de programação, configuração e implementação de serviços de redes distribuídas. Estes são, nem mais, nem menos, que os sucessores da unificação de muitas tecnologias de sistemas distribuídos utilizadas por muitos programadores na implementação de aplicações distribuídas, sobre a plataforma Windows, na passada década [WCFServices, 2011]. Como principais vantagens sobre os ASMX identificam-se as seguintes:

- Mais e melhores opções de segurança;
- Maior facilidade a nível de configuração e implementação;

- Funcionamento autónomo, ao contrário dos ASMX que necessitam do IIS para funcionar, os WCF têm um funcionamento autónomo que pode ser implementado através de uma aplicação de consola ou mesmo um serviço do Windows [HostingWCFServices, 2011].

Ao analisarmos as vantagens dos *WCF Services* em relação aos típicos *Web Services* (ASMX) e tendo em conta a sua escalabilidade e as opções avançadas de segurança que estes disponibilizam, a solução adoptada (em situações em que seja necessário estabelecer canais de comunicação) é a implementação de *WCF Services*.

Por outro lado, e já no contexto da transferência e sincronização de dados, os *Linked Servers* surgem como ferramentas capazes de estabelecer uma comunicação entre SGBD. A configuração de *Linked Servers* permite ao MS SQL Server executar comandos em servidores remotos, através de origens de dados OLE DB. Este tipo de configuração, para além de permitir o acesso remoto a servidores, possibilita também o tratamento de diversas fontes de dados de forma semelhante. Não menos importante é a capacidade de executar comandos, consultas distribuídas, actualizações e transacções sobre as fontes de dados heterogenias existentes [LinkingServers, 2011].

Para o acesso a dados do sistema SAP serão utilizadas as técnicas e ferramentas desenvolvidas e disponibilizadas por este. Desta forma, o desenvolvimento de soluções para integração entre os sistemas externos e o SAP deve utilizar sempre que possível as formas de interoperacionalidade que o SAP propõe, garantindo assim a manutenção e estabilidade das interfaces construídas. Embora o SAP disponibilize um leque de técnicas e ferramentas sobre este assunto, a Visabeira Digital definiu um processo para o acesso ao sistema SAP por aplicações externas. Este processo passa pela utilização da ferramenta *SAP Business Connector* (SAP BC) [SAPBC, 2011], que permite o acesso aos objectos de negócio do SAP utilizando a rede (Intranet ou Internet) como plataforma de comunicação. A ilustração seguinte apresenta o esquema de ligações que o SAP BC permite.

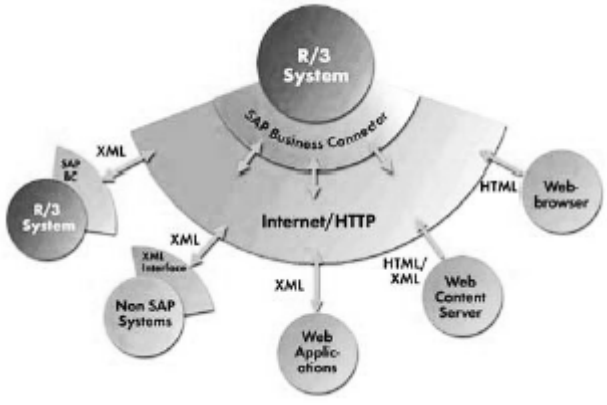

Ilustração 25 – SAP Business Connector.
(Fonte: http://www.erpgenie.com/sap/mysap/bus_connector.htm)

Utilizando o protocolo HTTP e documentos cuja estrutura é definida em XML, é efectuada a troca de informação entre os sistemas externos e o SAP. O XML é utilizado para definir uma estrutura comum a todos os documentos do negócio, incluindo as interfaces normalizadas e independentes da plataforma, designadas por *Business Application Programming Interfaces* (BAPIs). É

através das BAPIs que as aplicações externas conseguem aceder aos Objectos de Negócio SAP. A criação dos Objectos de Negócio e das BAPIs é da responsabilidade do SAP. Como vantagens para a sua utilização apresenta-se o facto de a sua interface estar normalizada, assim como a garantia dada pela empresa SAP da estabilidade das BAPIs entre diferentes versões do ERP. Uma alternativa a este processo era a manipulação directa dos dados armazenados na Base de Dados do SAP. Contudo, esta alternativa poderia ser muito perigosa (face ao relacionamento entre as entidades e à falta de validação dos dados), podendo pôr em causa a manutenção do SAP e a incompatibilidade entre futuras versões do ERP.

Como ferramenta de sincronização de dados capaz de permitir a configuração dos dados que se pretendem sincronizar, e ainda definir o modo como a sincronização deve ser feita, apenas uma única ferramenta foi identificada: a Microsoft Sync Framework [MSSyncFramework, 2011].

Esta ferramenta é uma plataforma de sincronização simples que permite a colaboração e o acesso *offline* por aplicações, serviços ou dispositivos. Possui tecnologias e ferramentas que permitem o acesso e a partilha de dados. Através desta *framework* facilmente podem ser criados ecossistemas que interligam qualquer aplicação com recurso a dados de diferentes tipos de armazenamento (ficheiros, SGBD, etc.), utilizando qualquer protocolo sobre qualquer rede. Uma vez que a *framework* está modelada de acordo com as APIs de acesso a dados do ADO.NET, o modo que esta fornece para sincronizar dados é muito intuitivo [MSSyncFramework, 2011].

4.2 Modelo lógico de dados

A fase de estudo e implementação do modelo de dados foi das mais complexas e críticas do projecto, uma vez que nesta solução o modelo de dados é o responsável não só pelo funcionamento do sistema, mas também pela sua expansibilidade a outras funcionalidades e áreas de negócio. Com base nesta característica, identificou-se a necessidade de dividir o modelo de dados em dois tipos de entidades, são elas:

- Entidades que têm sempre a mesma estrutura de dados ao longo do tempo, independentemente das funcionalidades e acessos do utilizador;
- Entidades responsáveis pelo armazenamento de dados específicos a cada funcionalidade.

A solução para implementar estes dois tipos de entidades passou pela criação de entidades com estrutura rígida, acrescidas em alguns casos de campos com tipo de dados XML, capazes de armazenar dados com estruturas diversas. O XML é um tipo de dados que suporta o armazenamento de dados com ou sem estrutura definida (*schema*). Uma vez que as colunas do tipo XML são armazenadas como objectos binários de grandes dimensões é de extrema importância a criação de índices. Os índices permitem aumentar quer o desempenho, quer o processamento nas operações sobre os dados XML. Para tal, o MS SQL Server permite a criação de dois tipos de índices, primários e secundários. Os índices primários indexam todos os nós, valores e caminhos de todas as instâncias XML de uma coluna. Já os índices secundários permitem a indexação específica dos dados XML. Existem três tipos de índices XML secundários, são eles: caminhos (*path*), propriedades (*properties*) e valores (*values*), que beneficiam o desempenho das consultas baseadas em caminhos XML, propriedades e valores, respectivamente [XMLBestPratices, 2011]. Também o acesso e manipulação de dados podem ser feitos através das inúmeras funções *xQuery* e *XPath* que o MS MSQL Server disponibiliza sobre este tipo de dados [KLEIN, 2006].

Através da implementação descrita anteriormente, o modelo lógico de dados proposto é capaz de armazenar informação, mesmo que a sua estrutura de dados seja completamente distinta e não esteja prevista, garantindo assim a característica expansível do sistema.

O problema imediato que surgiu com este projecto, foi a emissão de Guias de Transporte no modo *offline*. Atendendo a esta questão, pensou-se nas entidades que deveriam garantir e assegurar a operacionalidade do Sistema Cliente no modo *offline*. Surgiram assim as seguintes entidades:

- AuxiliarData – responsável pelo armazenamento local da informação de tabelas de apoio (materiais, fornecedores, depósitos, etc.) necessárias para que as funcionalidades possam operar no modo *offline*;
- CacheLogos – é a entidade principal do modo *offline*. Esta armazena a informação gerada pelas funcionalidades no modo *offline* e, posteriormente, sincroniza-a com o Sistema Integrador/Servidor, após o Sistema Cliente entrar no modo *online*;
- ConfigSynchronization – embora esta entidade não esteja directamente ligada ao modo *offline*, é a responsável por activar o mecanismo que garante a actualização dos dados que serão utilizados no modo *offline*, nomeadamente a entidade "AuxiliarData". Du-

rante o processo de actualização (no modo *online*) recorre à entidade "ConfigCache", criada com o intuito de armazenar a informação necessária para estabelecer uma ligação ao Sistema Integrador/Servidor;

- UsersAccessInfo – esta entidade foi definida com o intuito de possibilitar que, no modo *offline*, a aplicação informática existente no Sistema Cliente possa autenticar os seus utilizadores, caso contrário não poderiam utilizar a aplicação. Assim sendo, a entidade "UsersAccessInfo" armazena localmente, na Base de Dados do Sistema Cliente, a informação relativa à autenticação dos utilizadores. Cada registo desta entidade corresponde às credenciais de acesso de um determinado utilizador. Esta informação é armazenada após autenticação *online* no *Active Directory* do Grupo Visabeira. Assim sendo, e uma vez que esta informação só é necessária, no Sistema Cliente, para garantir a autenticação em modo *offline*, esta entidade não carece de sincronização com o Sistema Integrador/Servidor.

Identificadas e esquematizadas as entidades de suporte ao modo *offline* do Sistema Cliente, avançou-se para a necessidade de garantir a expansão do sistema proposto, a novas funcionalidades, empresas ou áreas de negócio. Com este objectivo foram enumeradas as seguintes entidades:

- ConfigFunctionalities – a entidade que disponibiliza as funcionalidades que são carregadas pela aplicação informática presente no Sistema Cliente;
- ConfigUsers – responsável por armazenar a informação específica dos utilizadores de um determinado Sistema Cliente, tais como: Nome, Número de funcionário, Empresa de Serviço, entre outros;
- ConfigUsersFunctionalities – é a entidade que determina quais as funcionalidades a que um determinado utilizador possui acesso;
- ConfigUsersOrganizationAccess – entidade responsável por determinar os acessos de um utilizador, aos dados da organização. Foi definido pela Visabeira Digital que o acesso a dados teria em conta a estrutura organizacional na qual os dados se enquadram, isto é, os dados serão definidos tendo em conta os campos: Firma, Organização de Compras, Centro Logístico e Depósito. Desta forma, será esta entidade a responsável por definir qual a estrutura organizacional a que um determinado utilizador tem acesso. Tomemos por exemplo, o utilizador A definido como os seguintes valores: Firma = Viatel, Organização de Compras = Viatel, Centro Logístico= Rede Fixa, Depósito = Viseu. Quer isto dizer que o utilizador A tem acesso a toda a informação (materiais, fornecedores, etc.) do depósito de Viseu, do centro logístico da Rede Fixa pertencente à firma e organização de compras da Viatel. Ficou definido que estes acessos não seriam dependentes de qualquer funcionalidade.

Por último, ficaria apenas em falta o requisito, imposto pela Viatel, de garantir um canal de comunicação que permitisse enviar alertas/notificações para os técnicos. Sobre este assunto foi definida uma nova entidade, denominada "CacheOperations". Esta entidade, para além do envio de alertas e notificações para o Sistema Cliente, permite ainda assegurar um canal de envio de informação para outras funcionalidades existentes no Sistema Cliente, alargando assim o âmbito da sua existência. Este processo é descrito mais à frente.

Após o enquadramento das principais entidades do sistema proposto, a Ilustração 26, agora apresentada, mostra o modelo lógico de dados proposto para este projecto.

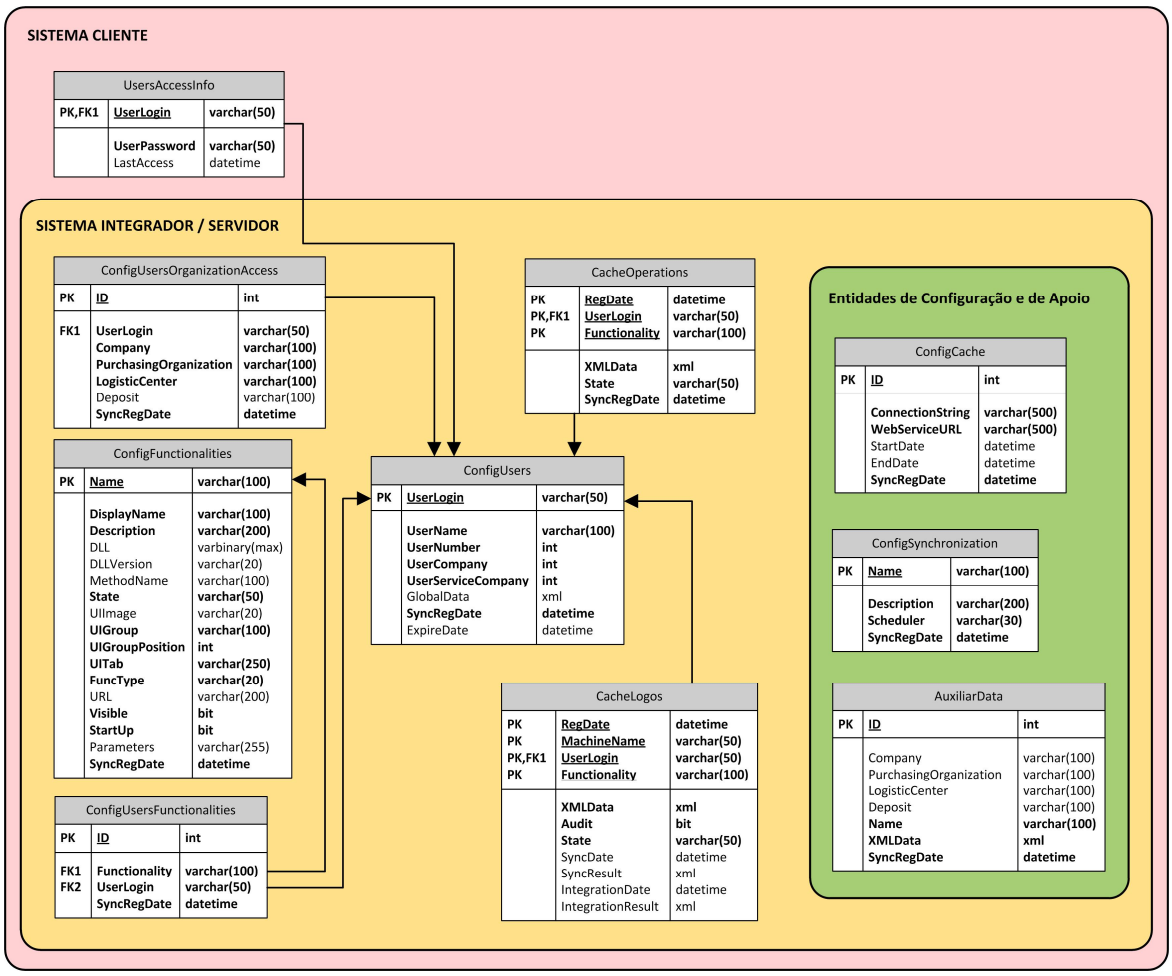

Ilustração 26 – Modelo lógico de dados proposto.

Como podemos ver na Ilustração 26, o modelo lógico de dados encontra-se agrupado em três áreas:

- Uma área de "Entidades de Configuração e de Apoio", representada na ilustração a cor verde;
- Uma área denominada de "Sistema Integrador/Servidor", onde estão as entidades que o Sistema Integrador/Servidor terá implementado na sua Base de Dados e sincronizará com o Sistema Cliente (inclusive as entidades referidas no ponto anterior);
- Por fim, uma área referente ao Sistema Cliente. Como é visível na Ilustração 26 este sistema engloba todas as entidades anteriormente descritas, mais a entidade designada por "UsersAccessInfo".

De todas as entidades representadas no modelo lógico de dados, apenas as tabelas comuns aos dois sistemas (Cliente e Integrador/Servidor) é que são sincronizadas com o auxílio do mecanismo de sincronização.

Todas as entidades que são alvo de sincronização incluem um campo "SyncRegDate". Este campo indica a data e hora referente ao momento em que a informação foi actualizada dos Sistemas de Informação Externos para o Sistema Integrador/Servidor.

Embora o diagrama da Ilustração 26 não seja especialmente complexo (em comparação, por exemplo, com um modelo lógico de dados de um qualquer ERP), a utilização comum de entidades pelos vários sistemas importa num aumento de complexidade, especialmente com a aludida inclusão de campos do tipo de dados XML, para acomodar a diversidade de dados que terão de armazenar. Assim, ir-se-á descrever primeiramente cada uma das entidades de configuração e de apoio, nomeadamente as entidades "ConfigCache", "ConfigSynchronization" e "AuxiliarData".

Entidade ConfigCache

O objectivo desta entidade é armazenar informação relativa ao Sistema Integrador/Servidor para que o Sistema Cliente saiba como estabelecer um canal de comunicação. A sua estrutura é apresentada na ilustração seguinte.

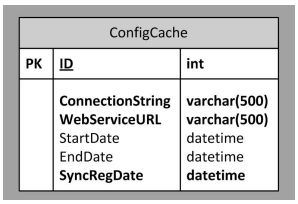

Ilustração 27 – Estrutura da entidade ConfigCache.

O campo "ConnectionString" é o responsável por armazenar os dados que permitem estabelecer o acesso à Base de Dados existente no Sistema Integrador/Servidor. Já o campo "WebServiceURL" contém o endereço *Web* para o Web Service genérico que permite o "Acesso Directo a Dados" dos Sistemas de Informação Externos, introduzido no subsecção 3.1.1. Os campos "StartDate" e "EndDate" indicam o intervalo de datas para as quais o registo é válido e a respectiva ligação pode assim ser activada. Estas datas permitem a definição de vários registos para períodos temporais distintos. O campo "SyncRegDate", como anteriormente descrito, representa a data e hora em que o sincronismo ocorreu no Sistema Integrador/Servidor.

Uma das possibilidades que a entidade "ConfigCache" permite, é a rápida migração para outro Sistema Integrador/Servidor, isto é, se houver a necessidade de mudar de Sistema Integrador/Servidor, apenas é necessário actualizar a informação desta tabela para que o Sistema Cliente automaticamente passe a comunicar com o novo Sistema Integrador/Servidor.

Entidade ConfigSynchronization

A entidade ConfigSynchronization é aquela que contém a informação sobre a periodicidade de sincronização dos diversos dados. O principal objectivo para a implementação desta entidade, é o de permitir a criação de agendamentos ao minuto e facilmente configurável para diversos tipos de dados (tabelas de apoio, funcionalidades, etc.). Com este objectivo em mente, surge a estrutura apresentada na Ilustração 28.

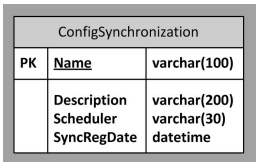

Ilustração 28 – Estrutura da entidade ConfigSynchronization.

Nesta entidade cada registo corresponde a um agendamento a ser executado pelo mecanismo de sincronização.

O campo "Name" além de ser chave primária é também o responsável por identificar os dados que o mecanismo de sincronização irá sincronizar. Por exemplo, o registo referente ao agendamento das funcionalidades tem como valor o texto "ConfigFunctionalities". Quando o mecanismo de sincronização é desencadeado, estabelece uma ligação ao Sistema Integrador/Servidor e envia como parâmetro o valor do campo "Name". Desta forma, o Sistema Integrador/Servidor sabe que está perante um pedido de sincronização de funcionalidades, e, mediante outra informação disponibilizada (tal como o nome do utilizador) devolve as funcionalidades respectivas, caso existam novas versões. Assim sendo, este campo funciona como uma espécie de filtro que será aplicado sobre os dados que serão sincronizados e que permitirá ao Sistema Integrador/Servidor determinar qual a informação que deverá ser sincronizada.

Já o campo "Description" apenas contém uma mera descrição da informação que será sincronizada aquando o mecanismo de sincronização utilizar o respectivo registo.

O campo "Scheduler" representa a periodicidade de sincronização através de uma estrutura semelhante ao agendamento CRON, existente em Sistemas Operativos Linux. Este encontra-se repartido em cinco grupos separados por espaços devidamente identificados na Ilustração 29.

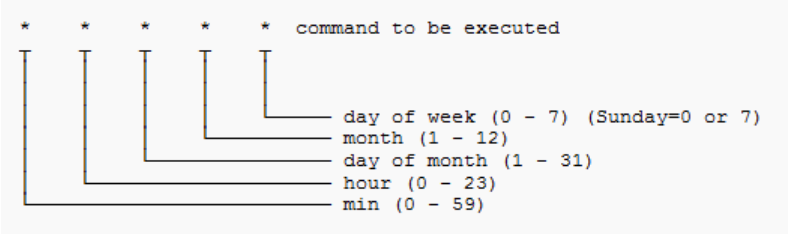

Ilustração 29 – Estrutura para criação de agendamento segundo o CRON.
(Fonte: http://en.wikipedia.org/wiki/Cron)

O primeiro grupo representa o minuto, entre zero e cinquenta e nove, para o qual será executada a sincronização. O segundo grupo representa a hora. O terceiro grupo é o responsável por definir o dia do mês para a qual a sincronização está activa (valores entre um e trinta e um). O quarto grupo indica o mês. E, por fim, o quinto grupo que indica o dia da semana. Para cada grupo para além dos intervalos de valores numéricos em cima identificados, é possível utilizar algumas configurações avançadas recorrendo aos seguintes caracteres específicos (, - / *).

Vírgula (,)

O caractere vírgula (,) define uma lista de valores para os quais a sincronização fica activa. Por exemplo, se utilizarmos os valores 1,2,3 no quinto campo (referente ao dia da semana), significa que o sincronismo irá ocorrer a todas as segundas, terças e quartas-feiras.

Hífen (-)

O caractere hífen (-) pode ser utilizado para definir um intervalo de valores. Por exemplo, o valor 12-16 no segundo campo (referente às horas) significa que o sincronismo irá acontecer entre as 12 horas e as 16 horas, isto é, nas horas 12, 13, 14, 15 e 16.

Barra (/)

O caractere barra (/) serve para incrementar o valor base definido. Com este caractere conseguimos, por exemplo, definir uma periodicidade para todas as horas ímpares bastando somente preencher o segundo campo com o seguinte valor: **1/2**. Significa que o sincronismo ocorre na hora 1 e sucessivamente de duas em duas horas, ou seja, hora 1,3,5,7, etc.

Asterisco (*)

O caractere (*) é o responsável por aceitar qualquer valor. Por exemplo, se utilizarmos este caractere no quarto campo (campo referente ao mês) significa que o sincronismo ocorrerá todos os meses.

Conjugando os quatro caracteres em cima apresentados, o leque de combinações possíveis é enorme, permitindo configurar e parametrizar as periodicidades das sincronizações de acordo com as necessidades. Por exemplo, podemos facilmente definir sincronizações para ocorrerem entre as 8 horas da manhã e as 19 horas da tarde, em intervalos de meia hora, todas as segundas, quartas e sextas-feiras para todos os meses, através do valor: **0/30 8-19 * * 1,3,5**.

Entidade AuxiliarData

A entidade "AuxiliarData" é aquela que recebe toda a informação referente a tabelas de apoio necessárias a algumas funcionalidades. A informação de cada tabela de apoio é convertida e armazenada no formato XML, permitindo assim o armazenamento dos dados das várias tabelas de apoio numa única entidade. Esta informação é posteriormente convertida em tabelas temporárias, na Base de Dados do Sistema Cliente, no momento em que as funcionalidades são carregadas. O objectivo da criação das tabelas temporárias é libertar recursos (memória do sistema) fazendo uso da elevada capacidade e desempenho do MS SQL Server. Quer isto dizer que, em vez de carregar a informação de tabelas de apoio para variáveis de memória existentes na aplicação informática, são criadas tabelas temporárias no MS SQL Server que serão depois utilizadas pelas respectivas funcionalidades.

Para classificar e identificar a informação de cada tabela de apoio ao nível da organização foram adicionados os campos "Company", "PurchasingOrganization", "LogisticCenter" e "Deposit", visíveis na ilustração seguinte.

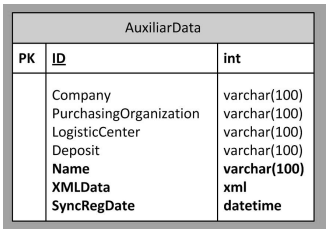

Ilustração 30 – Estrutura da entidade AuxiliarData.

Cada túpulo da entidade "AuxiliarData" representa uma tabela de apoio, que é identificada pelo campo "Name". Esta entidade é também capaz de armazenar tabelas de apoio transversais à aplicação informática, como por exemplo, a tabela de traduções. Para identificar estas tabelas de apoio, todos os campos de classificação organizacional, acima indicados, estão vazios.

Já o campo "XMLData" é o responsável por armazenar todo o conteúdo de uma tabela de apoio no formato XML. Esta entidade permite que o sistema apresentado tenha uma flexibilidade extra, uma vez que caso uma dada funcionalidade necessite de mais uma tabela de apoio, apenas será necessário disponibilizar essa mesma tabela nesta entidade. Todo o modelo lógico de dados permanece inalterável e rapidamente a tabela fica disponível no Sistema Cliente.

Analisando o modelo lógico de dados proposto na Ilustração 26, verificamos que a entidade "AuxiliarData" não está relacionada com nenhuma outra tabela. Deste modo, as tabelas de apoio são identificadas por um nome e pela sua estrutura organizacional (Firma, Organização de Compras, Centro Logístico e Depósito). Assim sendo, a estrutura organizacional será a ligação entre as tabelas de apoio disponíveis e os acessos que um determinado utilizador possui (entidade "ConfigUsersOrganizationAccess"). Com o objectivo de minimizar o número de tabelas a sincronizar entre o Sistema Integrador/Servidor e o Sistema Cliente, o mecanismo de sincronização de dados existente no Sistema Cliente, apenas sincroniza as tabelas de apoio para os utilizadores autenticados e de acordo com os seus acessos, configurados na entidade "ConfigUsersOrganizationAccess".

Entidade ConfigUsers

Esta entidade está directamente relacionada com toda a informação armazenada sobre o utilizador.

Um dos requisitos do Sistema Cliente identificado pela Visabeira Digital é o suporte multilíngua. Esta característica é configurável por utilizador. Com o intuito de permitir que o utilizador se autentique num outro Sistema Cliente e obtenha o ambiente previamente configurado e personalizado por ele, a informação específica a cada utilizador é armazenada e sincronizada. Contudo, a língua adoptada não é a única informação específica ao utilizador, no contexto da emissão de Guias de Transporte, ficou definido que a numeração da Guia seria no formato "Número do funcionário – Número sequencial". Como tal é também necessário o armazenamento do número da última guia emitida pelo utilizador. Para resolver esta situação, foi adicionado um campo "GlobalData" do tipo XML que permite o armazenamento dinâmico de toda a informação específica do utilizador. A Ilustração 31 mostra um exemplo para o valor do campo "GlobalData" relativamente a uma configuração da língua portuguesa e com um número total de 329 Guias de Transporte emitidas.

```
<GlobalData>
  <Row>
    <Propriedade>NumeroGuiasEmitidas</Propriedade>
    <Valor>329</Valor>
  </Row>
  <Row>
    <Propriedade>Lingua</Propriedade>
    <Valor>PT</Valor>
  </Row>
</GlobalData>
```

Ilustração 31 – Armazenamento de informação específica ao utilizador.

Como podemos observar na Ilustração 26, existe um campo de nome "ExpireDate". Este campo é o responsável por indicar a data até à qual o utilizador poderá aceder à aplicação informática, no modo *offline*. A actualização deste campo acontece quando o acesso à aplicação informática expirou e, no modo *online*, é efectuada a autenticação com sucesso. Durante este processo é gerada uma nova data até à qual o utilizador poderá voltar a utilizar a aplicação no modo *offline*.

Entidade ConfigUsersFunctionalities

Como podemos analisar na ilustração seguinte, esta entidade apenas vê a sua existência pelo simples facto de estabelecer a relação entre os utilizadores e as funcionalidades.

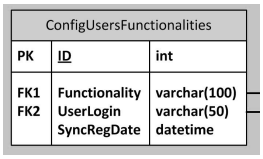

Ilustração 32 – Estrutura da entidade ConfigUsersFunctionalities.

Serve portanto para definir os acessos a funcionalidades por parte de um determinado utilizador. Cada utilizador poderá ter acesso a mais que uma funcionalidade. Do mesmo modo, a mesma funcionalidade será distribuída por um conjunto de utilizadores. Estamos perante um relacionamento "muitos-para-muitos" de onde surgiu esta entidade. Esta relação entre os utilizadores e as suas respectivas funcionalidades permite ainda a redução dos dados que serão sincronizados, quer isto dizer que para um determinado Sistema Cliente apenas serão sincronizadas as funcionalidades a que o utilizador desse Sistema Cliente possui acesso.

Entidade ConfigUsersOrganizationAccess

A entidade "ConfigUsersOrganizationAccess" foi criada com o objectivo de atribuir acessos a nível da organização a determinado utilizador. Por norma, cada utilizador apenas possui acesso a um armazém, podendo haver um número muito reduzido de utilizadores com acesso a mais que um armazém. Por exemplo, o UtilizadorA pertence à empresa Viatel, opera na Rede Móvel de telecomunicações e tem acesso ao armazém (em SAP configurado como depósito) de Viseu. A entidade "ConfigUsersOrganizationAccess" conteria o seguinte registo: *UserLogin* = UtilizadorA, *Company*=0027-Viatel, *PurchasingOrganization*=0027-Viatel, *LogisticCenter*=0272-Rede Móvel e *Deposit*=1001-Viseu.

Entidade ConfigFunctionalities

À semelhança da entidade anterior, as funcionalidades são configuradas numa plataforma *Web* em SharePoint. Devido ao funcionamento dual que a arquitectura permite, cada funcionalidade é classificada como sendo *Online* ou *Offline*. Em ambos os casos existe um conjunto de informação comum que deve ser registada, tal como:

- O nome único da funcionalidade (Name);
- O nome que será visível na aplicação (DisplayName);
- Uma breve descrição da mesma (Description);
- O estado (campo "State", valores: Desenvolvimento, Qualidade e Produção);
- O nome da imagem associada à funcionalidade (UIImage);
- O nome do grupo e posição do mesmo na barra de menus (UIGroup e UIGroupPosition);
- O nome do separador no qual o grupo se encontra inserido (UITab);
- O tipo de funcionalidade ("FuncType" com o valor *online* ou *offline*);
- A indicação se a funcionalidade está visível ao utilizador (Visible);
- A indicação se a funcionalidade inicia automaticamente após arranque da aplicação (StartUp)
- E, por último, alguns parâmetros de entrada para a funcionalidade (Parameters). Estes parâmetros permitem que as funcionalidades, devidamente codificadas, possam operar de diferentes modos de acordo com o valor dos parâmetros de entrada.

Caso a funcionalidade seja do tipo *online,* significa que esta utiliza exclusivamente o acesso *online* a dados.

Para funcionalidades do tipo *offline*, esta para além de ter um comportamento *online* é também capaz de operar no modo *offline*.

Nas situações em que o campo "URL" está preenchido, a aplicação do Sistema Cliente, ao carregar a funcionalidade, irá apresentar um *Web Browser* embutido disponibilizando o conteúdo do URL configurado. Esta característica permite organizar no Sistema Cliente, um ambiente de trabalho onde o colaborador disponha de todas as funcionalidades que necessita para trabalhar, permitindo assim, o carregamento também de páginas *Web*, como por exemplo os *sites* das soluções BeOn anteriormente apresentadas. Este *Web Browser*, à semelhança de outros, é capaz de desempenhar todo o tipo de acções, tais como: a navegação entre páginas, a execução de scripts (JavaScript, CSS, etc.), entre outros.

Para funcionalidades com o campo *DLL* preenchido, o mecanismo de actualização do Sistema Integrador/Servidor faz o *download* da DLL, anexada ao respectivo registo no portal SharePoint, e armazena o seu conteúdo (binário) em Base de Dados no campo "DLL". É ainda necessário indicar o método de arranque da funcionalidade (MethodName), isto é, o método que permite o arranque da funcionalidade depois de esta ter sido instanciada pela aplicação existente no Sistema Cliente.

Entidade CacheLogos

A entidade "CacheLogos" é a principal desta arquitectura. É esta que, no Sistema Cliente, armazena a informação gerada quando este está no modo *offline*, sincronizando-a posteriormente com o Sistema Integrador/Servidor para que possa ser integrada com o respectivo Sistema de Informação Externo. Ao analisarmos a estrutura apresentada para esta entidade, através da Ilustração 33, verificamos que a sua chave primária é composta pelos campos "RegDate", "MachineName", "Userlogin" e "Functionality".

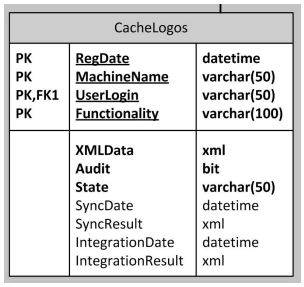

Ilustração 33 – Estrutura da entidade CacheLogos.

Uma vez que a chave primária permite referenciar de forma unívoca um registo numa entidade [DAMAS, 2005], quer isto dizer que um registo nesta entidade é único e identificado pela data e hora (incluí milissegundos) do seu registo (campo "RegDate"), pelo nome da máquina onde está a ser executada a aplicação informática (campo "MachineName"), pelo colaborador que criou o registo (campo "UserLogin") e pela funcionalidade originária (campo "Funcionality"). Uma vez que com estes campos conseguimos identificar um registo, o campo responsável por armazenar toda a informação gerada pela funcionalidade é o campo "XMLData", do tipo XML. Mais uma vez o tipo de dados XML permite-nos uma grande flexibilidade no armazenamento de dados com múltiplas estruturas. A Ilustração 34 apresenta o exemplo da informação de uma Guia de Transporte gerada pela funcionalidade de emissão de Guias de Transporte, e armazenada no formato XML.

```
<MovimentosArtigos>
  <CabecalhoGeral>
      <FirmaCodigo>0027</FirmaCodigo>
      <OrganizacaoComprasCodigo>0027</OrganizacaoComprasCodigo>
      <TipoMovimento>415</TipoMovimento>
      <NGuia>S5869-328</NGuia>
      <DataDocumento>2011-07-06 20:03:58</DataDocumento>
      <DataLancamento>2011-07-06 20:04:15</DataLancamento>
      <Matricula />
      <Referencia />
      <CategoriaMateriais>Novos</CategoriaMateriais>
  </CabecalhoGeral>
  <CabecalhoOrigem>
      <CentroLogisticoCodigo>0271</CentroLogisticoCodigo>
      <CentroLogisticoDesignacao>Viatel Rede Fixa</CentroLogisticoDesignacao>
      <DepositoCodigo>1000</DepositoCodigo>
      <DepositoDesignacao>PT Porto</DepositoDesignacao>
      <ElementoPEPCodigo />
      <ElementoPEPDesignacao />
      <ElementoPEPFirmaFuncionario />
      <PontoDescarga />
      <ElementoPEPNFuncionario />
      <CodigoFornecedor />
  </CabecalhoOrigem>
  <CabecalhoDestino>
      <CentroLogisticoCodigo>0271</CentroLogisticoCodigo>
      <CentroLogisticoDesignacao>Viatel Rede Fixa</CentroLogisticoDesignacao>
      <DepositoCodigo />
      <DepositoDesignacao />
      <ElementoPEPCodigo>0027.10002566</ElementoPEPCodigo>
      <ElementoPEPDesignacao />
      <ElementoPEPFirmaFuncionario>27</ElementoPEPFirmaFuncionario>
      <ElementoPEPNFuncionario>3956</ElementoPEPNFuncionario>
      <PontoDescarga>-</PontoDescarga>
      <PontoDescargaFornecedor />
      <CodigoFornecedor />
  </CabecalhoDestino>
  <Artigos>
    <Artigo>
      <Linha>1</Linha>
      <Codigo>100003456</Codigo>
      <Quantidade>44</Quantidade>
      <NSerie />
      <Designacao>LAMPADA STD CLARA 60W</Designacao>
      <UnMedida>UN</UnMedida>
      <Observacoes />
      <TipoMaterial>ZROH</TipoMaterial>
      <TipoAvaliacao />
      <ExisteEmStock>False</ExisteEmStock>
    </Artigo>
  </Artigos>
  <Assinatura>
    iVBORw0KGgoAAAANSUhEUgAAAlgAAACWCAYAAAACG/YxAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwA
  </Assinatura>
</MovimentosArtigos>
```

Ilustração 34 – Exemplo da estrutura XML de uma Guia de Transporte.

Como podemos ver na Ilustração 34 a estrutura XML está dividida em cinco áreas distintas, são elas:

- CabecalhoGeral – esta área armazena informação geral da Guia de Transporte;
- CabecalhoOrigem – contém a informação que indica a origem do movimento desta guia (técnico, depósito, fornecedor, etc.), ou seja, a entidade de onde saíram os materiais;
- CabecalhoDestino – contém a informação que indica o destino do movimento desta guia (técnico, depósito, fornecedor, etc.), ou seja, a entidade que procedeu à recolha dos materiais;
- Artigos – esta área contém discriminadamente a informação dos artigos (materiais) que foram movimentados;

- Assinatura – Este último campo é o responsável por armazenar a informação da assinatura digital recolhida pelo técnico. A informação é armazenada no formato de StringBase64.

O campo "Audit", presente nesta entidade, é responsável por indicar se o registo em causa contém algum tipo de erro que deverá ser auditado. Cada vez que é detectado um erro na aplicação, não só é alertado o utilizador como também é inserido um registo nesta entidade descriminando o erro ocorrido e sumarizando os passos que estavam a ser executados. Este tipo de registo é direccionado para um Sistema de Informação Externo, criado especificamente para receber os erros das várias aplicações, para que possam ser posteriormente analisados e corrigidos pela equipa de manutenção e desenvolvimento de *software* da Visabeira Digital.

Entidade CacheOperations

A existência da entidade "CacheOperations" tem como objectivo permitir o envio de informação para um ou vários utilizadores. Na prática é a implementação de um canal que permite o envio de notificações e mensagens para os utilizadores. Para que esta entidade não ficasse restrita apenas a esta funcionalidade, acrescentou-se o campo "Functionality". Desta forma, esta entidade fornece um canal unidireccional entre o Sistema Integrador/Servidor e o Sistema Cliente. No Sistema Cliente cada funcionalidade analisa e interpreta a informação que lhe pertence, identificada pelo campo "Functionality". Sobre este processo, para além da funcionalidade que apresenta as notificações/mensagens aos utilizadores, foi também implementada uma funcionalidade que executa código SQL directamente na Base de Dados local de cada Sistema Cliente. O objectivo desta funcionalidade é enviar e executar blocos de código SQL em todos ou em determinados Sistemas Clientes, permitindo efectuar alterações sobre a Base de Dados e sobre os seus registos de forma remota.

Toda a informação é transferida em formato XML através do campo "XMLData". De seguida é apresentado um modelo de uma notificação em formato XML.

```
<Table>
  <Row>
    <ID>2</ID>
    <Title>Bem vindo ao Logos, a nova aplicação da marca BeOn
    de suporte à logistica do Grupo Visabeira. Visabeira Digital</Title>
    <StartDate>2011-06-01T00:00:00</StartDate>
    <EndDate>2011-06-03T00:00:00</EndDate>
    <Created>2011-06-01T15:46:29</Created>
  </Row>
</Table>
```

Ilustração 35 – Exemplo do formato XML de uma notificação.

A estrutura do XML poderá ser completamente distinta uma vez que o seu destinatário (funcionalidade para a qual é destinado o XML) saberá como interpretar o seu conteúdo. Como foi dito anteriormente, cada funcionalidade apenas analisa e interpreta a informação que lhe é destinada, identificando essa informação através do valor do campo "Functionality".

Apresentado todo o modelo de dados comum aos dois sistemas (Sistema Integrador/Servidor e Sistema Cliente), falta apenas introduzir a entidade "UsersAccessInfo".

Entidade UsersAccessInfo

Esta entidade apenas existe no Sistema Cliente e é a responsável por armazenar a informação referente aos utilizadores do Sistema Cliente, nomeadamente a sua informação de autenticação (UserLogin *e* UserPassword), e a data e hora do seu último acesso (LastAcess). O valor da *password* do utilizador é encriptado utilizando o algoritmo Triple DES (3DES) [ANDERSON, 2008]. De entre muitos algoritmos de encriptação existentes, foi escolhido o Triple DES uma vez que este é o implementado nas aplicações da Visabeira Digital. Por razões de segurança, o processo de autenticação nunca desencripta a *password* armazenada, em vez disso encripta o valor da *password* introduzida no ecrã de *login* e compara-o com o valor encriptado e armazenado na Base de Dados, validando assim o acesso à aplicação.

4.3 Sistema Cliente

O Sistema Cliente é o responsável por garantir uma boa interacção com o utilizador através da sua aplicação informática. De igual modo, foi concebido de forma a ser robusto e garantir sem qualquer tipo de falhas todos os objectivos propostos. Para uma fácil compreensão da implementação deste sistema, a sua descrição foi subdividida em duas partes distintas. Uma parte é referente à implementação da aplicação informática e a outra é a implementação do mecanismo de sincronização dos dados existentes na Base de Dados local. Por fim, é ainda apresentado um levantamento sobre os requisitos mínimos recomendados para a utilização do Sistema Cliente descrito nesta secção.

4.3.1 Aplicação informática

A aplicação informática é aquela que estabelece o contacto directo com o utilizador. O seu funcionamento irá afectar a confiança e satisfação que o utilizador manifesta, ao utilizar a aplicação. Tendo esta ideia sempre em mente, toda a sua implementação teve em atenção o público-alvo, evidenciando sempre a usabilidade das suas funcionalidades. Os princípios de usabilidade foram aplicados não só a nível dos conteúdos apresentados, tais como, o número máximo de itens nos menus, a apresentação de formulários e textos, mas também a nível da interacção com o utilizador, mantendo um constante e elucidativo *feedback* com o utilizador após o desenrolar de cada operação [NIELSEN, 1993].

Na análise efectuada sobre as necessidades da aplicação informática identificaram-se cinco operações principais, indicadas no diagrama de casos de uso apresentado na Ilustração 36, que serão explicadas detalhadamente no desenrolar desta subsecção.

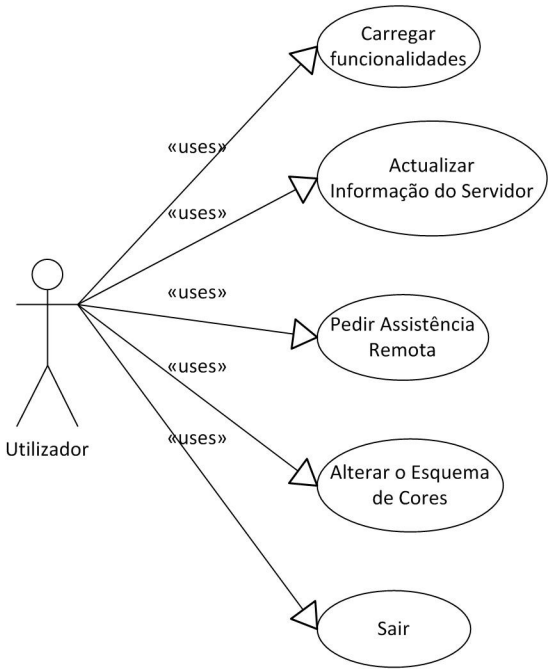

Ilustração 36 – Diagrama de Casos de Uso para as operações base da aplicação informática.

O utilizador, ao clicar no ícone da aplicação, inicia o seu processo de arranque. Este processo é composto por vários passos e regras de validação da aplicação. Ao iniciar a aplicação, esta verifica de imediato se o mecanismo de sincronização e o SGBD MS SQL Server está activo. Caso não esteja activo, o Sistema Cliente não é capaz de desempenhar as principais funções de manutenção, actualização e sincronização de dados, pelo que o utilizador é alertado da situação e o arranque da aplicação é cancelado. Caso esteja tudo operacional, é apresentado ao utilizador o ecrã de autenticação (Ilustração 37) onde este terá que introduzir as suas credenciais de acesso, as mesmas que utiliza na organização, e a língua na qual pretende visualizar os conteúdos disponibilizados pela aplicação.

Ilustração 37 – Ecrã de autenticação da aplicação informática.

Para simplificar este processo, o campo "Utilizador" é preenchido de forma automática com o último acesso válido na aplicação. Como podemos analisar na ilustração acima apresentada, a aplicação foi baptizada com o nome Logos (proximidade com logística uma vez que foi esta a neces-

sidade imediata do projecto). No campo superior direito, aparece o logótipo BeOn que apadrinhou esta solução.

Relativamente à implementação dos diversos idiomas, e uma vez que o sistema poderá operar no modo *offline*, não era possível a utilização de um motor de traduções *online*. Assim sendo, foi criada uma tabela de apoio responsável por armazenar as diferentes traduções para cada descrição/frase apresentada pelas funcionalidades. Efectuado o *login*, cada vez que uma funcionalidade é apresentada ao utilizador, esta verifica primeiro o idioma seleccionado e recorre depois à tabela de apoio para traduzir o seu conteúdo.

Após o arranque da aplicação e se for a primeira vez que o utilizador se autentica na aplicação, é necessário que esta esteja no modo *online*. Neste cenário, a aplicação tenta estabelecer uma ligação à *active directory* do Grupo Visabeira, com o intuito de validar se as credenciais introduzidas existem, e são válidas no sistema. Caso as credenciais sejam válidas, a aplicação procede à encriptação e ao armazenamento local das mesmas, de forma a permitir futuros acessos quer no modo *online* quer no modo *offline*.

Um outro passo existente no processo de autenticação, requisito deste projecto, é o mecanismo de segurança que bloqueia o acesso à aplicação a um de determinado utilizador. A implementação deste mecanismo mostrou-se muito simples, uma vez que para além das credenciais de acesso de cada utilizador, é também armazenada informação relativamente ao seu último acesso ao sistema. Com esta informação, o mecanismo de segurança verifica se a data actual é superior à data limite de acesso à aplicação. Para proteger eventuais alterações da data e hora do sistema local, o mecanismo valida sempre a data actual em função da data do último acesso do utilizador, de forma a garantir que o relógio não foi atrasado. A aplicação é desbloqueada quando um utilizador com o acesso bloqueado efectua a autenticação na aplicação e esta tem acesso à rede, ou seja, encontra-se no modo *online*. Nesta situação o Sistema Integrador é informado do estado da aplicação e emite uma nova data limite de acesso, campo “ExpireDate” armazenado na entidade “ConfigUsers” conforme Ilustração 26.

Após autenticação com sucesso, e apenas quando esta funciona no modo *online*, a aplicação envia um pedido para o Sistema Integrador/Servidor indicando o estado *online* do utilizador. Este pedido actualiza directamente o estado do utilizador no portal de configuração da arquitectura. A ilustração abaixo apresentada, mostra alguns utilizadores e o seu respectivo estado.

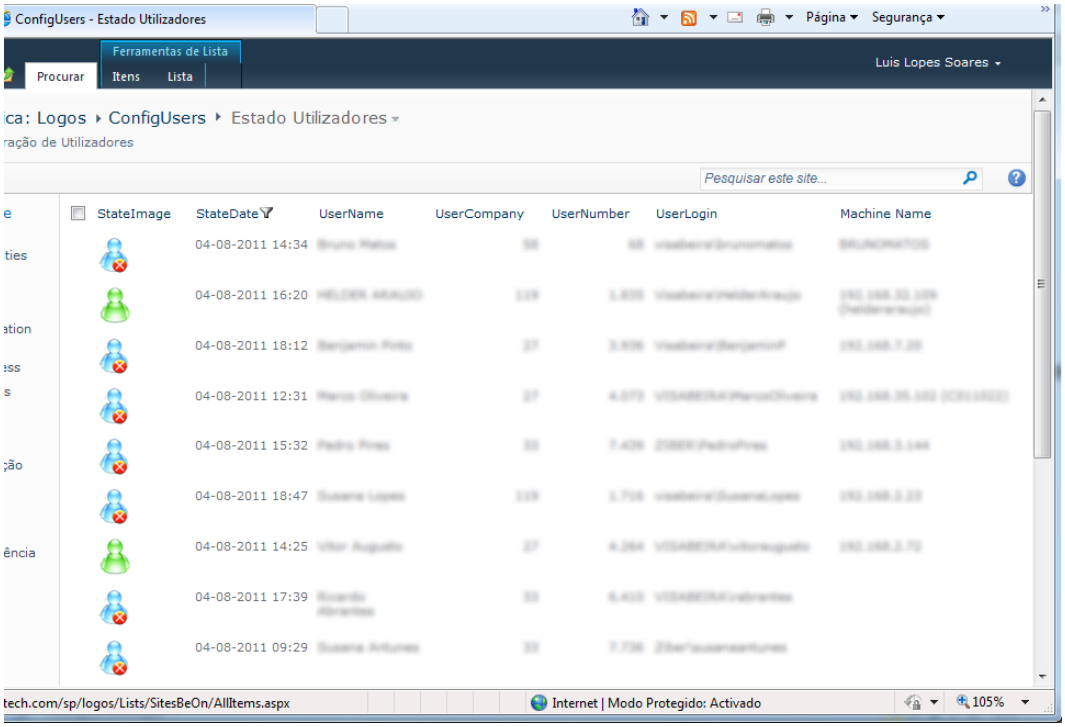

Ilustração 38 – Estado dos utilizadores do Sistema Cliente.

Em paralelo, a aplicação carrega todos os menus com as respectivas funcionalidades às quais o utilizador possui acesso. O ambiente gráfico da aplicação informática encontra-se subdividido em quatro áreas principais, visíveis na Ilustração 39.

Ilustração 39 – Ambiente gráfico da aplicação informática.

A primeira área (área 1) representa a barra de menus. Esta é responsável pela apresentação e organização dos botões que representam as diversas funcionalidades. As funcionalidades apresentadas nesta área variam de acordo com os acessos de cada utilizador. A sua disposição é agrupada por separadores e grupos à semelhança do friso do Microsoft Office 2010, o que permite uma mai-

or proximidade por parte dos utilizadores, devido ao conhecimento já adquirido nas ferramentas Office. Quando o utilizador clica num botão, a aplicação obtém o binário ou URL da funcionalidade que está armazenado na Base de Dados local, e carrega-o de forma dinâmica para a área 2 visível na ilustração. Abaixo apresenta-se o código que permite o carregamento e a apresentação dinâmica de funcionalidades, através do binário referente a uma DLL.

```
'Carregar dinamicamente a informação da assembly
Dim asm As Reflection.Assembly = Reflection.Assembly.Load(drFunctionality("DLL"))
'Objecto que irá conter o formulário
Dim oRes As Object = Nothing
'Procurar dinamicamente o tipo que contém o método definido na funcionalidade
For Each tpAsm As Type In asm.GetTypes
    If Not tpAsm.GetMethod(drFunctionality("MethodName")) Is Nothing Then
        'Criar uma instância para o tipo em causa
        oRes = asm.CreateInstance(tpAsm.FullName, True)
        'Invocar dinamicamente a função de arranque
        oRes.GetType.GetMethod(dsData.Tables(0)(0)("MethodName")).Invoke(oRes, New Object() {
                                                                New DataLayer(_BD_CONNECTIONSTRING, True), Me, _
                                                                New Parameters(strName, strDispName, htParam)})
        'Acelerar o processamento
        Exit For
    End If
Next
```

Ilustração 40 – Carregamento e apresentação dinâmica de funcionalidades.

Ainda na área 1, existe um pequeno botão no canto superior direito que permite minimizar a barra de menus maximizando a área útil para apresentação das funcionalidades (área 2).

A área 2 é aquela que apresenta as funcionalidades escolhidas pelo utilizador. O utilizador pode carregar várias funcionalidades ao mesmo tempo, contudo a mesma funcionalidade só será apresentada uma única vez. Por cada funcionalidade carregada é criado um separador. Caso a funcionalidade seleccionada já esteja carregada, após clique no respectivo menu esta será apresentada em primeiro plano. Uma vez que a aplicação informática permite a execução simultânea de várias funcionalidades, cada funcionalidade implementa as suas diversas operações no formato multitarefa. Este formato permite a execução de código em paralelo, aumentando assim o desempenho da funcionalidade e impedindo que esta bloqueie a interacção do utilizador com as restantes funcionalidades. Neste projecto, a implementação das operações multitarefa foi efectuada através de *BackgroundWorkers*, uma classe auxiliar disponível na Framework .NET [ALBAHARI, 2011]. Com o intuito de permitir uma maior flexibilidade e comodidade ao utilizador, as funcionalidades podem ser desagregadas da aplicação e/ou organizadas de vários modos (vertical, horizontal, lado a lado, minimizada, etc.) conforme apresenta a ilustração seguinte.

Ilustração 41 – Organização e arranjo das funcionalidades.

Se analisarmos a Ilustração 41, notamos a existência de pequenos quadrados azuis com uma seta. Estes quadrados azuis aparecem quando o utilizador arrasta uma funcionalidade para fora do separador. Recorrendo a estas setas o utilizador poderá voltar a embutir a funcionalidade na aplicação, mas na posição e da forma que pretender (vertical, horizontal, lado a lado, fixada na lateral, etc.).

Quando a aplicação inicia, é também carregada uma funcionalidade base com o logótipo BeOn e uma grelha de notificações/avisos destinados ao utilizador. Esta funcionalidade está visível na Ilustração 39.

Já a área 3 disponibiliza uma grelha denominada "Mensagens de aplicação". Nesta grelha são apresentadas mensagens das diversas funcionalidades, permitindo um constante *feedback* com o utilizador, quer do seu histórico de acções, quer das operações que as diversas funcionalidades estão executar.

Por último, a quarta área (área 4) é a que disponibiliza ao utilizador alguma informação adicional, apresentada na ilustração seguinte.

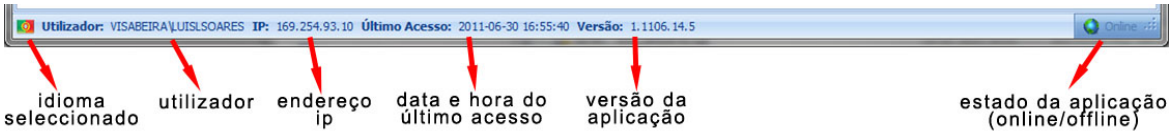

Ilustração 42 – Base de estados da aplicação informática.

O ícone referente ao estado da aplicação indica ao utilizador se a aplicação é capaz de comunicar com o Sistema Integrador/Servidor ou não. Quando a aplicação muda para o estado *Offline*, o ícone ficará a cor cinzenta e todos os botões referentes a funcionalidades *Online* ficarão bloqueados, impedindo assim o seu acesso.

Ainda que não muito explícito existe um botão principal na aplicação, localizado no canto superior esquerdo da Ilustração 39, que disponibiliza ao utilizador quatro operações gerais sobre a aplicação informática, conforme é apresentado na Ilustração 43.

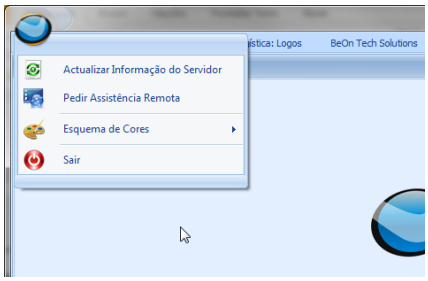

Ilustração 43 – Operações gerais da aplicação.

A opção "Actualizar Informação do Servidor", a primeira da ilustração, é a opção que permite ao utilizador a sincronização manual dos dados entre o Sistema Cliente e o Sistema Integrador/Servidor.

A segunda opção, "Pedir Assistência Remota", foi criada com o intuito de permitir aos utilizadores finais, solicitar o auxílio da equipa de *help-desk* com o intuito de esclarecer alguma dúvida que surja durante a utilização da aplicação informática. Esta opção só está disponível no modo *Online* e na Intranet do Grupo Visabeira devido às políticas de acesso externo. Ao activar esta opção, o Sistema Cliente envia um pedido de assistência remota por e-mail para a equipa de *help-desk*. Este pedido contém uma chave com toda a informação necessária para estabelecer uma ligação remota à máquina do utilizador, conforme apresentado na Ilustração 44.

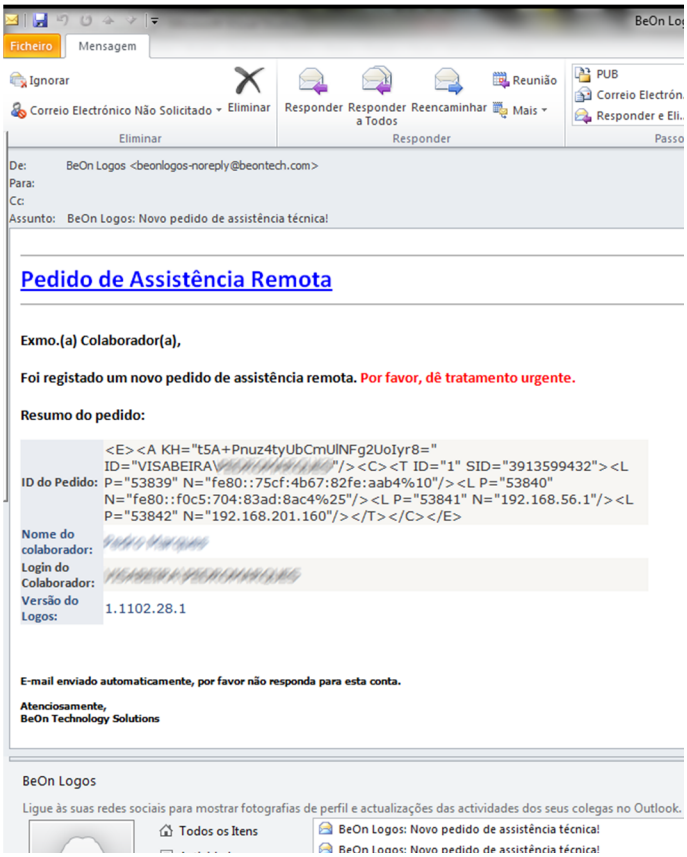

Ilustração 44 – Pedido de assistência remota por e-mail.

Após pedido de auxílio, a equipa de *help-desk* utiliza uma funcionalidade específica que permite estabelecer a ligação remota através da chave enviada. A ilustração seguinte mostra o processo de aceder remotamente a uma máquina. Na ilustração apresentada a máquina à qual foi estabelecido o acesso remoto tem a particularidade de ter dois monitores associados, pelo que a funcionalidade irá mostrar toda a sua área de trabalho.

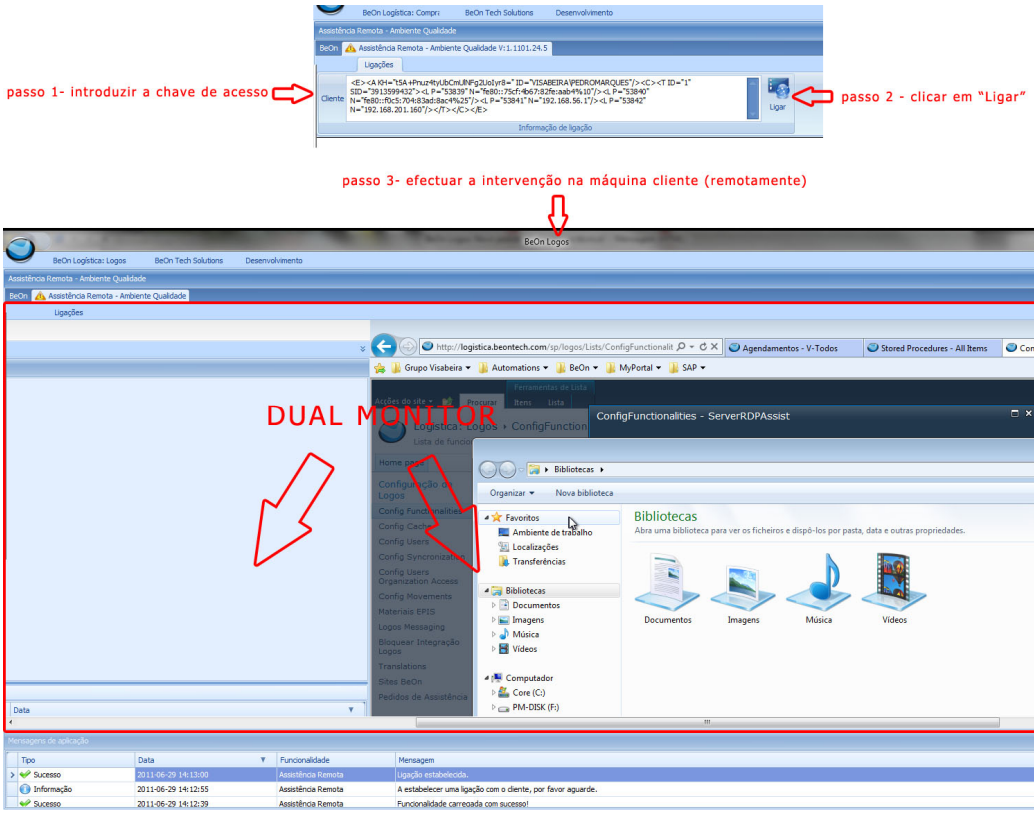

Ilustração 45 – Intervenção remota a uma máquina com dois monitores, de um técnico.

Tanto a chave como a ligação remota à máquina do utilizador são obtidas utilizando a livraria RDPCOMAPILib. Mais uma vez a escolha incidiu sobre a tecnologia Microsoft. A chave tem um carácter temporário e só é válida enquanto o utilizador não sair da aplicação informática.

Na terceira opção, "Esquema de Cores", é permitido ao utilizador alterar o esquema de cores de toda a aplicação informática, conforme Ilustração 46. O objectivo desta opção é simplesmente permitir que o utilizador personalize a aplicação à sua medida para que esta lhe proporcione um maior prazer e conforto na sua utilização.

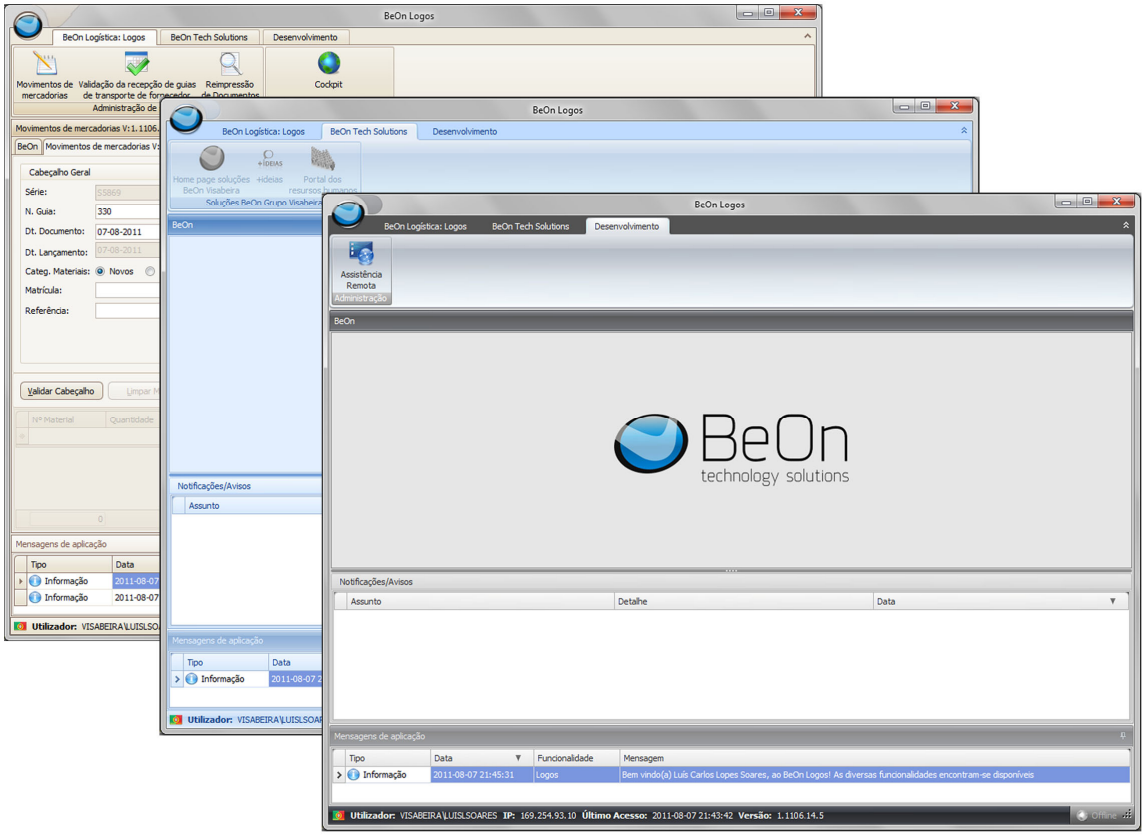

Ilustração 46 – Personalização de aspecto gráfico da aplicação.

A última opção visível na Ilustração 43 é, como o próprio nome indica, a responsável por finalizar a aplicação informática.

4.3.1.1 Funcionamento Dual

Neste projecto o funcionamento dual é sem dúvida uma característica única e a destacar. A implementação deste funcionamento passou pela separação das funcionalidades em dois grupos: funcionalidades *online* e funcionalidades *offline*. Quer as funcionalidades *online* quer as funcionalidades *offline* estão armazenadas localmente na Base de Dados do Sistema Cliente.

Relativamente às funcionalidades *Web* (campo "URL" preenchido) a informação armazenada é referente ao endereço da página *Web* que será apresentada num Web browser disponibilizado na própria aplicação informática. Estas são classificadas como funcionalidades exclusivamente *online*. Um exemplo da utilização deste tipo de funcionalidades acontece no acesso a plataformas *Web* já implementadas. Para tal, o botão representativo dessa funcionalidade permite o acesso directo pelos utilizadores sem que haja a necessidade de abrir outra aplicação ou decorar o respectivo endereço electrónico.

Para as funcionalidades com o campo "DLL" preenchido (Windows Forms), a informação armazenada na Base de Dados é a informação binária da DLL correspondente. É de relembrar que a DLL é convertida e armazenada em formato binário quando esta é disponibilizada no Sistema Integrador / Servidor. Posteriormente é sincronizada para os Sistemas Clientes respectivos, para

poderem ser carregadas e apresentadas ao utilizador. São estas funcionalidades as responsáveis por permitir a utilização do funcionamento dual da arquitectura. Estas acedem a dados de Sistemas de Informação Externos, no modo *online*, e no caso da falha de ligação com este sistema, adoptam o modo *offline* onde recorrem à interacção com a base de dados local para continuar a desempenhar as mesmas funções. Contudo, não é obrigatório que todas as funcionalidades Windows Forms implementem o funcionamento dual da arquitectura. Um caso prático é a funcionalidade de Gestão e Validação de Guias de Transporte. Esta apenas funcionará no modo *online*, através do qual obtém informação e procede à sua análise e tratamento. No modo *offline*, cada funcionalidade define a estrutura de dados XML que será responsável por armazenar e garantir todos os seus requisitos de informação (campos necessários para armazenar toda a informação).

Em síntese, a aplicação informática abordada nesta subsecção permite funcionalidades sob o formato de páginas *Web* e funcionalidades Windows Forms. As páginas *Web* operam exclusivamente no modo *online*, enquanto as funcionalidades Windows Forms são capazes de operar no modo *online, offline* e *misto* (estabelecendo um consenso entre operar no modo *online* ou no modo *offline)*. São estas funcionalidades mistas que permitem explorar e tirar verdadeiramente partido da arquitectura dual proposta neste projecto.

4.3.1.2 Funcionalidade Emissão de Guias de Transporte

A funcionalidade que permite a emissão de Guias de Transporte foi a impulsionadora deste projecto. O principal objectivo da implementação desta funcionalidade é simplificar o processo de emissão de Guias de Transporte, evitando assim a utilização das diversas transacções SAP por parte dos técnicos, e diminuir o tempo necessário para emitir uma Guia de Transporte.

Em SAP, mais propriamente no módulo MM, cada movimento tem uma transacção associada, ou seja, se pretendermos efectuar movimentos de materiais de um armazém para um técnico é uma transacção, se for de um técnico para um armazém é outra, se for entre armazéns já terá que ser uma terceira transacção, etc. Este processo ficou extremamente simplificado, como podemos ver na Ilustração 47.

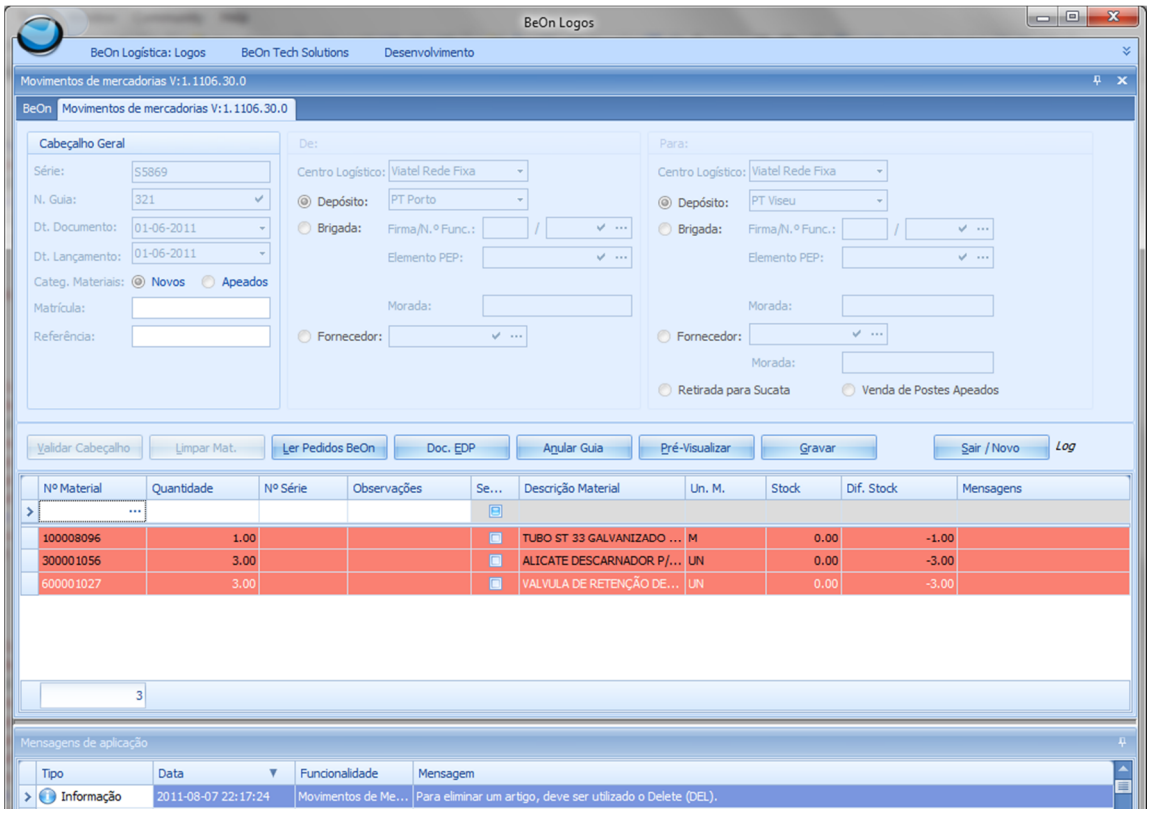

Ilustração 47 – Funcionalidade Emissão de Guias de Transporte.

Na funcionalidade aqui apresentada o técnico apenas terá que escolher a origem (área "De") e o destino (área "Para"). Mais tarde, será o Sistema Integrador o responsável por executar a respectiva transacção no módulo MM do SAP.

Com o objectivo de optimizar e diminuir o tempo necessário para a emissão de uma Guia de Transporte, foram implementados dois mecanismos:

- Registo de materiais recorrendo a um leitor de códigos de barras;
- Registo de materiais através do carregamento de pedidos de materiais efectuados pelos técnicos.

O primeiro ponto é identificado na operação de inserção de detalhes. Ao inserirmos um material do tipo seriado com o leitor de códigos de barras, a funcionalidade automaticamente preenche a informação do respectivo material. Se por acaso o leitor de códigos de barras não estiver disponível, a funcionalidade auxilia o registo de materiais seriados da seguinte forma:

1. O utilizador introduz o código do material;
2. O utilizador introduz a quantidade;
3. A funcionalidade verifica que é um material seriado e cria registos até perfazer a quantidade introduzida. Desta forma, o utilizador apenas terá que lançar números de série.

Relativamente ao segundo ponto, como foi abordado anteriormente, o técnico tem sempre a possibilidade de efectuar um pedido de material quando detecta carência de *stock* em certos materiais. Para tal recorre a uma solução BeOn que permite através do PDA efectuar o dito pedido de material. Neste contexto, caso a funcionalidade de emissão de Guias de Transporte esteja a operar

no modo *online*, a opção "Ler Pedidos BeOn" ficará activa. Esta opção permite carregar para a Guia de Transporte a informação de um Pedido de Material previamente seleccionado. Após este processo a Guia de Transporte fica pronta a emitir, aguardando apenas a validação das quantidades disponíveis em *stock* por parte do técnico.

Um outro requisito que surgiu com o intuito de reduzir a quantidade de papel gasta com a impressão de Guias de Transporte é a recolha da assinatura digital.

Esta assinatura é armazenada em sistema na própria Guia de Transporte. Assim sendo, de cada vez que uma guia é emitida, o técnico responsável pelo levantamento dos respectivos materiais assina-a através da respectiva funcionalidade e recebe uma via impressa já com a sua assinatura. A opção de assinar uma guia apenas fica disponível após a sua gravação. A ilustração seguinte mostra a funcionalidade responsável pela recolha e apresentação da assinatura.

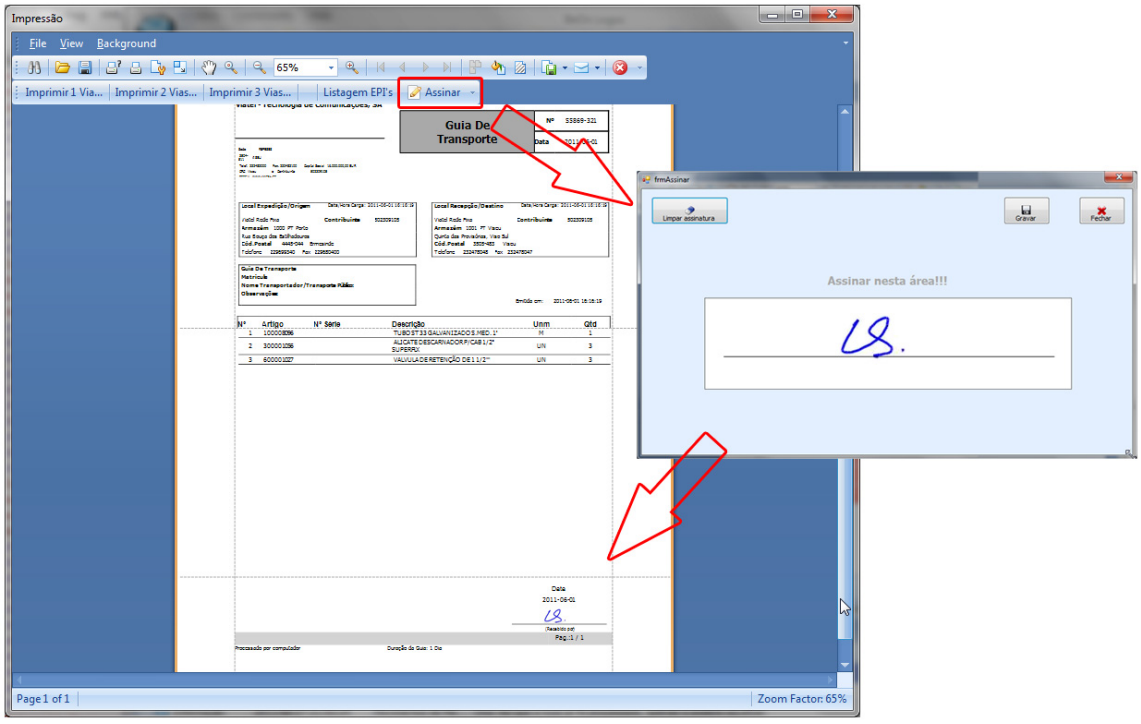

Ilustração 48 – Funcionalidade para assinar uma Guia de Transporte.

A assinatura é introduzida através de um dispositivo digital, nomeadamente uma caneta digital ou uma mesa digitalizadora. A informação referente à representação gráfica da assinatura é armazenada no próprio registo XML da Guia de Transporte.

4.3.1.3 Funcionalidade de Gestão e Validação de Guias de Transporte (cockpit)

O propósito desta funcionalidade é de, a qualquer momento e em tempo real, podermos consultar uma Guia de Transporte. Esta consulta é efectuada no modo *online* sobre a Base de Dados do Sistema Integrador/Servidor, local onde actualmente é armazenado todo o histórico das Guias de Transporte geradas pelo Sistema Cliente. A Ilustração 49 mostra a funcionalidade Cockpit, que permite a consulta sobre os dados existentes no Sistema Integrador/Servidor.

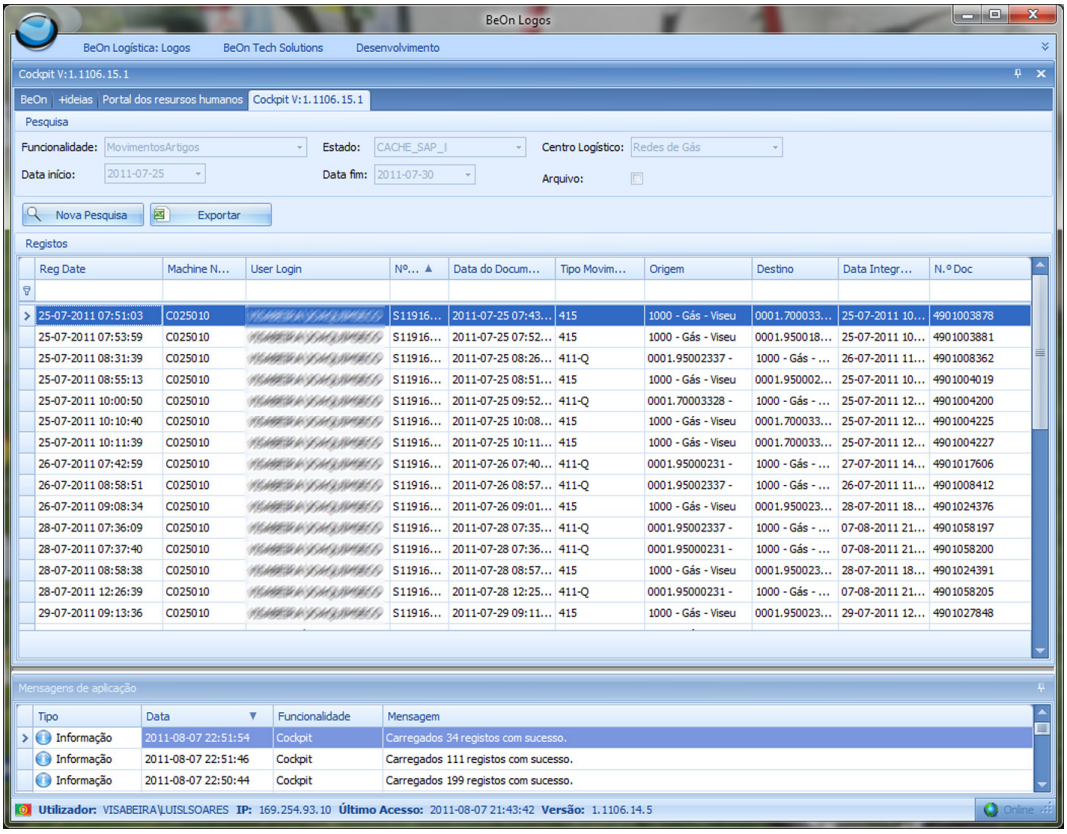

Ilustração 49 – Funcionalidade Cockpit.

Nesta funcionalidade, é possível detectar Guias de Transporte para a qual ocorreram erros de integração com o módulo MM do SAP (Estado = CACHE_SAP_I). Após correcção desses erros, pode ser feita nova tentativa de integração através do botão "Enviar para SAP", conforme apresenta a Ilustração 50.

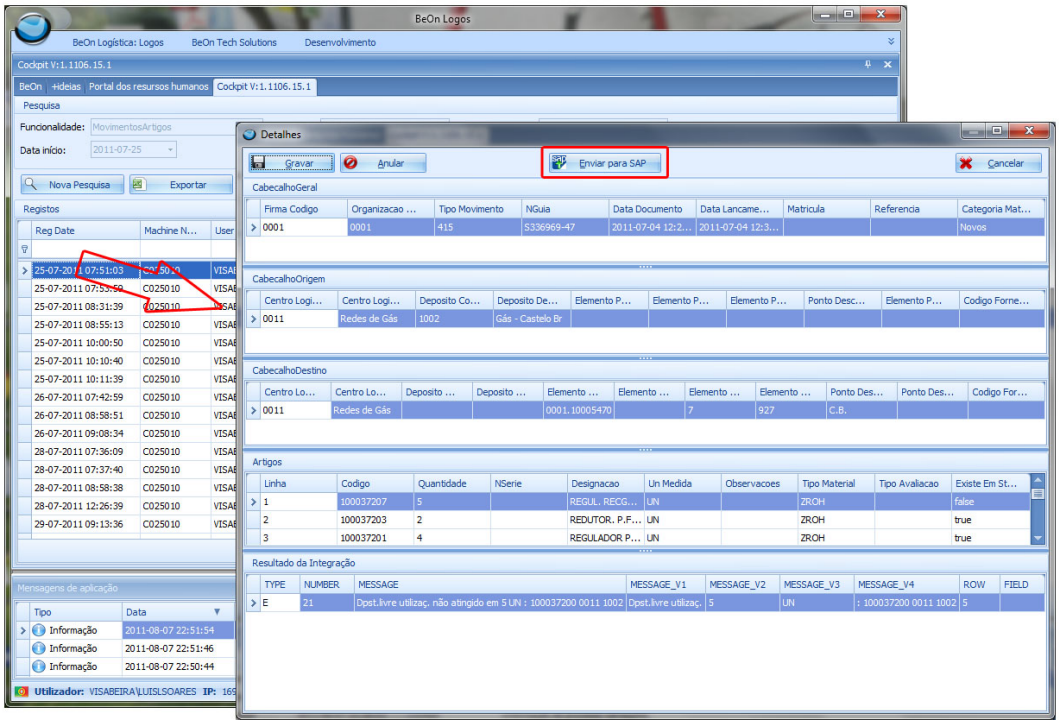

Ilustração 50 – Nova tentativa de integração de uma Guia de Transporte.

De igual modo, esta funcionalidade permite a consulta de Guias de Transporte integradas com sucesso directamente do sistema SAP, através da sua interface *Web*. A ilustração seguinte apresenta o acesso a um documento em SAP, através do botão "Consultar em SAP" disponível nesta funcionalidade.

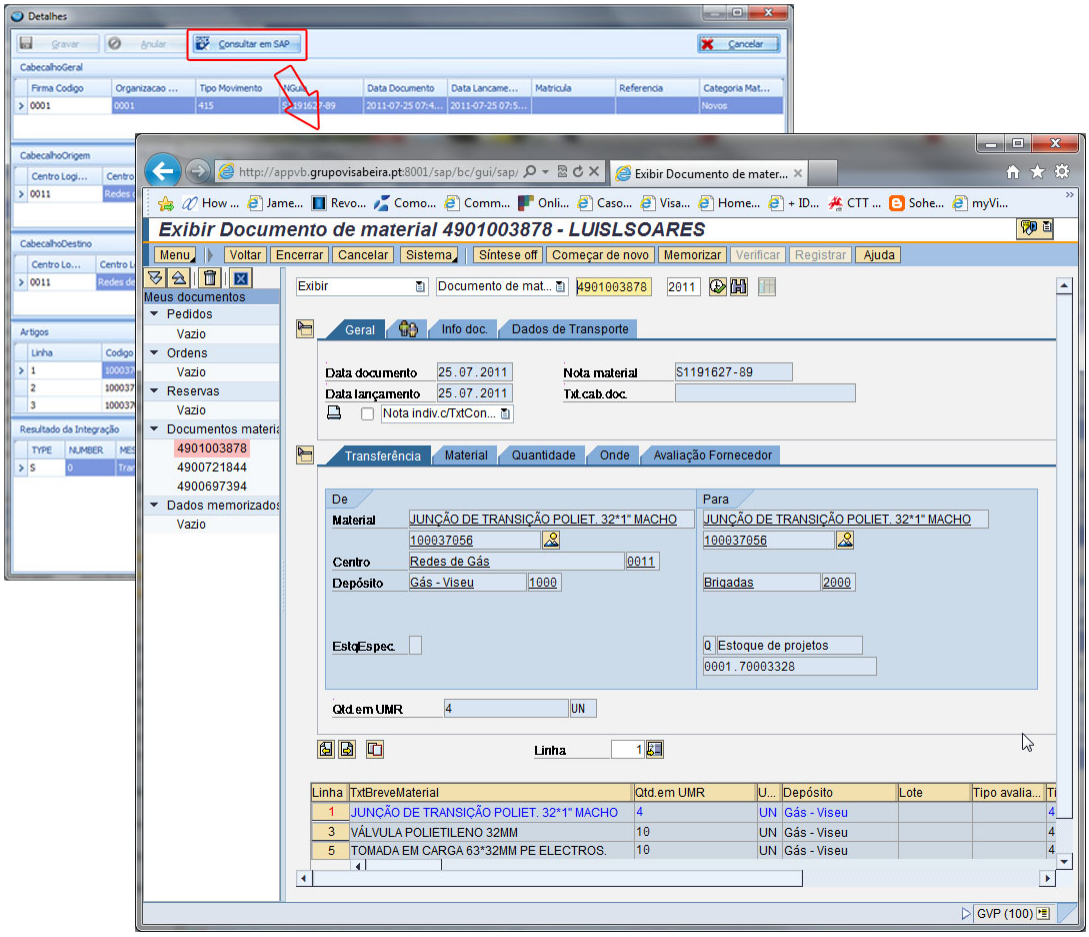

Ilustração 51 – Utilização da interface Web do SAP para consultar um documento.

4.3.2 Mecanismo de sincronização de dados

O mecanismo de sincronização de dados existente no Sistema Cliente desempenha um papel fundamental, uma vez que é este o responsável pela sincronização dos dados entre o Sistema Cliente e o Sistema Integrador/Servidor.

O objectivo deste mecanismo é a possibilidade de sincronizar dados de uma forma automática entre os sistemas anteriormente referidos, mesmo quando a aplicação informática não está activa. Desta forma, o seu funcionamento tem que ser independente da aplicação informática. Para criar esse sistema, apenas se identificaram as seguintes possibilidades:

- Desenvolver uma aplicação de consola (Console Application) e do seu agendamento através de uma tarefa agendada do Windows;
- Criar um serviço do Windows (Windows Service) onde o seu agendamento está implícito no mesmo.

Após análise sobre estas duas possíveis soluções, e uma vez que o Sistema Cliente será instalado por ClickOnce, a solução escolhida foi a criação de um serviço do Windows.

Para além da instalação do serviço do Windows ser totalmente integrada com o ClickOnce, não existe a necessidade de criar tarefas agendadas do Windows, como no caso da aplicação de consola. Outra mais-valia do serviço do Windows é o seu modo de arranque automático. Este modo de arranque permite que o serviço do Windows seja iniciado mesmo depois de a máquina ter sido desligada ou reiniciada. Por último, não necessita de acessos e configurações avançadas.

O mecanismo de sincronização activa-se a todos os minutos e verifica junto da entidade "ConfigSynchronization", introduzida na secção 4.2, se existem operações a sincronizar. Caso a hora actual do sistema valide algum dos agendamentos definidos nesta entidade, conforme a estrutura CRON atrás especificada, o mecanismo procede à sincronização dos respectivos dados entre o Sistema Cliente com o Sistema Integrador/Servidor.

Toda a informação sincronizada por este mecanismo, num determinado Sistema Cliente, é filtrada com base nos utilizadores que possuem *login* efectuado no Sistema Cliente, o qual está a sincronizar a informação. O objectivo da aplicação deste filtro é diminuir o volume de informação transmitida entre o Sistema Integrador/Servidor e o Sistema Cliente.

4.3.3 Requisitos de *hardware*

Como passo final da implementação do Sistema Cliente, surge a análise de requisitos de *hardware*. Para efectuar esta análise, fez-se uma recolha sobre os requisitos mínimos recomendados para cada componente que constitui o Sistema Cliente. Assim, procedeu-se à instalação de todos os componentes do Sistema Cliente sobre o sistema operativo Microsoft Windows XP, recorrendo-se depois à ferramenta Gestor de Tarefas do Windows para analisar os valores dos atributos "Utilização da CPU" e "Memória Física". Os valores médios obtidos durante a utilização do sistema permitiram-nos chegar a um consenso relativamente aos requisitos mínimos que a máquina deve suportar para que todo o Sistema Cliente funcione sem qualquer problema.

Como requisitos mínimos de *hardware* recomenda-se um processador com velocidade de 1.0GHz e memória RAM no valor de 512 MB.

A nível de requisitos de *software*, aconselha-se o sistema operativo Microsoft Windows XP ou superior.

4.4 Sistema Integrador/Servidor

Como vimos na secção anterior, o Sistema Cliente é aquele que estabelece o contacto directo com o utilizador final. Este sistema contém uma aplicação informática que permite ao utilizador a manipulação e inserção de novos dados. Posteriormente, através do seu mecanismo de sincronização, os novos dados são sincronizados entre as Bases de Dados do Sistema Cliente e do Sistema Integrador/Servidor.

Neste contexto, surge a presente subsecção onde se pretende abordar mais detalhadamente o Sistema Integrador/Servidor. Como o próprio nome indica, a principal função do Sistema Integrador/Servidor é a integração dos dados provenientes dos Sistemas Clientes com os Sistemas de Informação Externos. Esta integração apenas é possível graças ao mecanismo de actualização e integração de dados implementado na arquitectura proposta. Como função secundária, o Sistema Integrador/Servidor é também responsável por permitir o acesso directo a dados dos Sistemas de Informação Externos, quando este acesso for solicitado por funcionalidades do Sistema Cliente.

Este servidor, devido à sua grande importância na arquitectura proposta está protegido. Isto é, diariamente são efectuadas cópias de segurança para um servidor secundário de *Disaster Recovery* (DR) que arrancará caso seja detectada alguma anomalia no servidor primário.

4.4.1 Mecanismo de actualização e integração de dados

O mecanismo de actualização e integração de dados opera em dois processos distintos. Um é o processo de actualização de dados, onde o mecanismo actualiza os dados armazenados na sua Base de Dados local com os dados provenientes dos respectivos Sistemas de Informação Externos. Estes dados serão mais tarde sincronizados para os Sistemas Clientes. O processo de actualização de dados, aqui descrito, encontra-se representado na Ilustração 52 através do fluxo de informação "Fluxo 1".

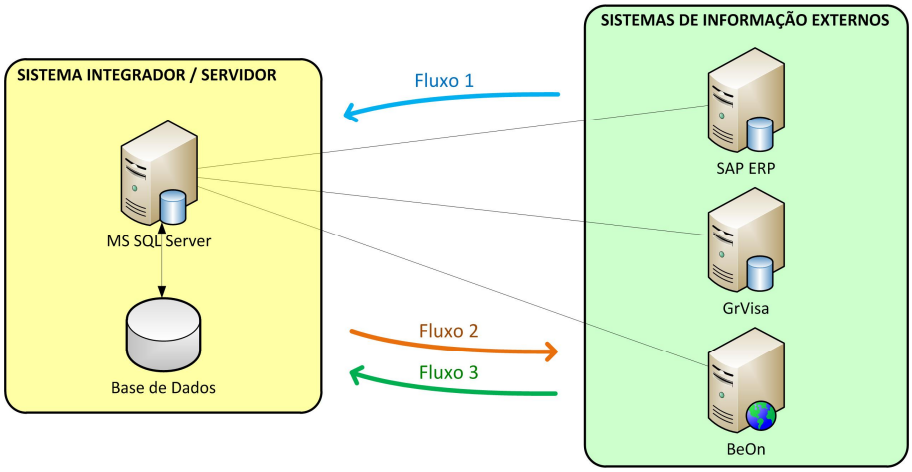

Ilustração 52 - Fluxos de informação do mecanismo de actualização e integração de dados.

O outro processo é o processo de integração de dados, no qual o mecanismo consulta a Base de Dados local ao Sistema Integrador/Servidor e selecciona os registos que estão por integrar. Para cada registo seleccionado, efectua a sua tentativa de integração com o respectivo Sistema de Informação Externa ("Fluxo 2" apresentado na ilustração anterior). O resultado proveniente da tentativa de integração (informação que contém os detalhes do sucesso ou erro da operação) é devolvido pelo fluxo de informação "Fluxo 3" da Ilustração 52 e armazenado no registo para o qual foi iniciado o processo de integração. Esta integração pode ser efectuada no modo síncrono (*on-time*) ou no modo assíncrono (através de rotinas de integração).

Em ambas as situações o mecanismo recorre às tecnologias e ferramentas identificadas na subsecção 4.1.3, implementadas de acordo com a descrição que se segue.

Durante a implementação deste mecanismo surgiram duas grandes questões, uma relativamente ao modo como este comunica com os diversos Sistemas de Informação Externos e outra relativamente ao modo como o mecanismo é activado.

4.4.1.1 Comunicação com os diversos Sistemas de Informação Externos

Relativamente a esta questão, e uma vez que o Sistema Integrador/Servidor tinha disponível o motor de Base de Dados MS SQL Server, optou-se por centralizar e efectuar todos os acessos aos Sistemas de Informação Externos, através do MS SQL Server. Os métodos de acesso utilizados serão descritos de seguida, nesta subsecção. De igual modo, é o MS SQL Server que define e implementa quer o método de acesso que utiliza para aceder a um determinado Sistema de Informação Externo, quer a lógica de negócio que aplica sobre os dados. Para simplificar este processo, definiu-se que para cada acesso (leitura ou escrita de dados) a um determinado Sistema de Informação Externo, é criado um procedimento armazenado com um nome sugestivo e identificador das operações que realiza.

A criação dos procedimentos armazenados ditos anteriormente, permitiu simplificar a implementação do mecanismo de actualização e integração de dados. Para compreendermos melhor a simplicidade do funcionamento deste mecanismo, de seguida apresenta-se o funcionamento base do mecanismo de actualização e integração de dados.

O mecanismo de actualização e integração de dados tem um comportamento análogo a uma rotina (*job)* SQL, quer isto dizer que o mecanismo é activado repetidamente em determinados intervalos de tempo (previamente configurados). Após a sua activação, este apenas executa um determinado procedimento armazenado que é o responsável por obter ou integrar um conjunto de dados com os respectivos Sistemas de Informação Externos. Por exemplo, no processo de actualização dos dados de apoio utilizados pelas funcionalidades que operam no modo *offline*, o mecanismo apenas invoca o procedimento armazenado denominado de "SincronizaMateriais". Este procedimento armazenado é o responsável por estabelecer a ligação ao SI SAP e actualizar a Base de Dados local com a informação devolvida, para que esta seja posteriormente sincronizada com os vários Sistemas Clientes. O processo de integração de dados com os Sistemas de Informação Externos é semelhante, ou seja, neste processo é invocado um procedimento armazenado que, mediante a informação que dispõe para integrar, determina o Sistema de Informação Externo e as transacções que deverão ser executadas. De igual modo, o procedimento armazenado é o responsável por estabelecer a ligação ao SI e efectuar todas as transacções necessárias para cumprir com sucesso a integração dos dados. No caso das Guias de Transporte, estas podem ser integradas com o módulo MM do SAP de dois modos distintos. Um através da integração *online*, onde é enviada a informação da Guia de Transporte para um procedimento armazenado responsável por efectuar a integração em tempo real. O segundo modo recorre ao mecanismo de integração aqui descrito, onde um outro procedimento armazenado recolhe a informação das Guias de Transporte não integradas (armazenadas na Base de Dados do Sistema Integrador/Servidor) e, para cada Guia de Transporte, tenta a sua integração.

Uma vez apresentado o funcionamento do mecanismo de actualização e integração de dados, descrevem-se de seguida os métodos utilizados para estabelecer o acesso e a transferência de da-

dos entre os diversos Sistemas de Informação Externos, que constituem a arquitectura proposta nesta dissertação.

Para o sistema GrVisa e algumas aplicações verticais com Base de Dados MS SQL Server, o acesso é garantido e simplificado através do uso de Linked Servers, conforme se apresenta na ilustração seguinte.

Ilustração 53 – Acesso a Sistemas de Informação Externos através de *Linked Server*.

Uma vez estabelecida a ligação com a respectiva Base de Dados remota, o *Linked Server* permite a manipulação dos dados (consulta, inserção, actualização, etc.), em entidades existentes na Base de Dados remota, como se se tratasse de uma Base de Dados local. Na subsecção 4.1.3, anteriormente apresentado, é explicado de uma forma mais aprofundada o uso e a aplicabilidade desta ferramenta de acesso a dados.

Já relativamente à ligação com o Sistema de Informação externo das soluções BeON, e uma vez que a informação é armazenada em listas SharePoint, o acesso a estas soluções é estabelecido através dos *Web Services* que o SharePoint disponibiliza, nomeadamente o *Web Service* "lists.asmx". O *Web Service* "lists.asmx" disponibiliza métodos de acesso a listas do SharePoint, apresentados anteriormente na Ilustração 15. Para todas as aplicações que não disponibilizem qualquer meio que permita o acesso a dados por sistemas externos, serão criados Web Services segundo as normas e a tecnologia .NET [BALLINGER, 2003].

Contudo, o MS SQL Server não permite estabelecer uma ligação directa com os sistemas que utilizam o *Web Service* como canal de comunicação (soluções BeOn e outros). De seguida apresentam-se as duas principais tentativas de interligação do MS SQL Server com os *Web Services*.

A primeira abordagem passou pela utilização de código Transact-SQL e dos procedimentos armazenados do sistema, nomeadamente os procedimentos armazenados referentes a automações *Object Linking and Embedding* (OLE) [OLE Automation, 2011]. As automações OLE permitem a comunicação entre processos, baseada em objectos *Component Object Model* (COM) introduzidos pela Microsoft. Assim, através da função "sp_OACreate", que permite a criação de uma instância de um objecto OLE, criou-se o objecto "MSSOAP.SoapClient30", objecto este que é o responsável por estabelecer uma ligação a um determinado *Web Service*. Após instanciação do objecto atrás referido, recorreu-se à função "sp_OAMethod" para invocar um método do objecto OLE, ou seja, na prática estamos a invocar um método do *Web Service*. Com estes passos, a comunicação através de *Web Services* é garantida. Contudo, este tipo de implementação era um pouco rígida e carecia de alguns cuidados de programação, nomeadamente na libertação dos recursos utilizados, uma vez que estas funções fazem parte de código que não é gerido pelo sistema operativo (*Unmanaged Code*) [Managed e Unmanaged Code, 2003]. Outra dificuldade encontrada traduziu-se na defini-

ção das credenciais de autenticação à rede e ao *Web Service*. Por estes motivos, esta solução não foi adoptada.

Após algumas pesquisas com ênfase na necessidade de consumir *Web Services* através do MS SQL Server, muitas delas remetiam para a implementação e utilização de funções *Common Language Runtime* (CLR), as quais viriam a ser adoptadas neste projecto para solucionar esta questão. O CLR permite o desenvolvimento de procedimentos, *triggers* e funções numa das linguagens CLR disponíveis, particularmente Microsoft Visual C# .NET, Microsoft Visual Basic.NET e Microsoft Visual C++. O código executado pelo CLR, ao contrário da primeira abordagem, é gerido pelo sistema operativo (*Managed Code*). Assim, quer a reserva, quer a libertação de recursos é garantida pelo sistema operativo, libertando o programador dessa responsabilidade. Outra particularidade do CLR é a de permitir a expansão da Base de Dados com novos tipos de dados, desde que devidamente programados [CLR, 2011].

O exemplo agora apresentado foi extraído de [CLRExternalWebService, 2011] e adaptado para o MS SQL Server 2008. Para podermos utilizar as funções CLR, é necessário proceder primeiro à sua activação e, uma vez que a Base de Dados irá aceder a recursos externos, importa elevar o nível de confiança da Base de Dados, isto é, configurar a Base de Dados de modo a que o MS SQL Server não limite qualquer operação executada por esta, nomeadamente a execução de funções CLR. A ilustração seguinte apresenta o código responsável por garantir as configurações acima indicadas.

```
sp_configure 'clr enabled', 1
GO
RECONFIGURE
GO

ALTER DATABASE Testes SET trustworthy on
```

Ilustração 54 – Código SQL para activar funções CLR e elevar o nível de confiança de uma BD.

Após executadas as configurações anteriores, os próximos passos para estabelecer uma ligação entre o MS SQL Server e um *Web Service* são os seguintes:

- Obter a informação que define o *Web Service*;
- Desenvolver código CLR responsável por consumir o *Web Service*;
- Gerar uma única DLL com o conteúdo dos dois pontos anteriores e uma segunda DLL com o respectivo código de serialização XML;
- Criar as Assemblies no MS SQL Server para as DLL geradas anteriormente;
- E, por fim, criar uma função SQL responsável por utilizar as Assemblies criadas.

Os cinco pontos acima mencionados serão agora detalhadamente apresentados e acompanhados por um exemplo prático. Neste exemplo, o MS SQL Server obtém a informação de um *Web Service* (TesteService.asmx), que contém apenas um método (HelloWorld) responsável por devolver o seguinte texto: “Hello World”. Comecemos então pela obtenção da informação que descreve o *Web Service*.

Tipicamente todo o conteúdo de um *Web Service* (inclusive os métodos que ele disponibiliza) é descrito numa linguagem própria que se apresenta no formato XML, designada por *Web Services Description Language* (WSDL). É esta informação que permite a instanciação e utilização quer do *Web Service*, quer dos métodos que este disponibiliza [LOPES e RAMALHO, 2005]. Uma aplicação que nos ajuda na obtenção desta informação tem o nome de "WSDL.exe" e está localizada na pasta "Microsoft SDKs\Windows\v7.0A\bin". Através da configuração de alguns parâmetros, aquando a sua invocação, podemos facilmente definir o ficheiro no qual será guardada a informação referente ao *Web Service*. Por exemplo, se invocarmos o seguinte comando "*wsdl.exe /o:TesteService.vb /l:VB /n:ClrWebServices http://localhost/TesteService.asmx*", a aplicação irá gerar um ficheiro com o nome "TesteService.vb" que possui uma classe pública na linguagem Visual Basic .NET de nome "TesteService", definida no namespace "ClrWebServices".

O passo seguinte é a codificação do bloco de código CLR responsável, neste exemplo, por invocar o *Web Service* e devolver a informação proveniente dessa mesma invocação no formato de texto. A Ilustração 55 apresenta uma função de nome "InvocaWebService" que como o próprio nome indica, é a responsável por invocar o método "HelloWorld" do *Web Service* "TesteService" e devolver o seu resultado no tipo de dados "String".

```
Imports ClrWebServices
Partial Class TesteClass
    Inherits System.Web.UI.Page

    ''' <summary>
    ''' Função que invoca e devolve uma string com o output do webservice
    ''' </summary>
    ''' <returns></returns>
    ''' <remarks></remarks>
    Public Function InvocaWebService() As String
        Dim _webservice As TesteService = New TesteService
        Dim _respostaWebService As String = _webservice.HelloWorld()
        Return _respostaWebService
    End Function
End Class
```

Ilustração 55 – Código CLR responsável pela invocação do Web Service "TesteService".

O código apresentado na ilustração anterior está armazenado num ficheiro com o nome "InvocaWebService.vb".

De seguida, pretende-se compilar a informação (WSDL) que descreve o *Web Service* e o código CLR, apresentados anteriormente, numa única DLL e criar ainda o respectivo código de serialização XML. A criação do código de serialização XML é necessária uma vez que o MS SQL Server não permite a serialização dinâmica de XML [CLR, 2011]. O comando "*csc /t:library InvocaWebService.vb TesteService.vb*" gera o ficheiro "InvocaWebService.dll" e o comando "*sgen /a: InvocaWebService.dll*" gera o ficheiro "InvocaWebService.XmlSerializers.dll", que contém o código de serialização XML. Após a criação destes dois ficheiros, podemos prosseguir para o passo seguinte.

O passo agora descrito refere-se à criação de duas Assemblies no MS SQL Server, uma para cada DLL gerada anteriormente. O código Transact SQL responsável pela criação das duas Assemblies é apresentado na ilustração seguinte.

```
CREATE ASSEMBLY ClrWebServices
FROM 'C:\ClrWebServices\InvocaWebService.dll'
WITH PERMISSION_SET = EXTERNAL_ACCESS
GO

CREATE ASSEMBLY [ClrWebServices.XmlSerializers]
FROM 'C:\ClrWebServices\InvocaWebService.XmlSerializers.dll'
WITH PERMISSION_SET = SAFE
GO
```

Ilustração 56 – Código Transact SQL para criação de Assemblies.

Por último, segue a criação de uma função responsável por efectuar o mapeamento de código SQL para a Assemblie. Na Ilustração 57 podemos verificar o código que permite a criação desta função.

```
CREATE FUNCTION InvocaWebService()
RETURNS varchar(max) WITH EXECUTE AS CALLER
AS
EXTERNAL NAME ClrWebServices.TesteClass.InvocaWebService
GO
```

Ilustração 57 – Criação de uma função SQL que invoca uma Assemblie.

Esta função, como podemos ver na ilustração atrás apresentada, devolve o resultado da sua invocação no tipo de dados "varchar(max)". Para completar o exemplo prático, apresenta-se na Ilustração 58 o código que permite a invocação e obtenção dos dados do *Web Service* utilizado neste exemplo.

```
SELECT dbo.InvocaWebService()
```

Results | Messages

	(No column name)
1	Hello World

Ilustração 58 – Invocação do *Web Service* "TesteService.asmx" através do MS SQL Server.

Embora o código CLR utilizado no exemplo anterior tenho sido reduzido, deu para perceber o grande dinamismo que este permite ao ser utilizado em conjunto com o MS SQL Server. Ainda no exemplo anterior, verificamos que em apenas cinco passos conseguimos estabelecer uma ligação do MS SQL Server a outro Sistema de Informação através de *Web Services*. Contudo, esta solução não era suficiente para sistemas de grande dimensão, uma vez que para cada *Web Service* utilizado teriam que ser executados todos os cinco passos anteriormente descritos. No caso do sistema BeOn, este disponibiliza um conjunto elevado de soluções *Web* representadas por mais de 20 *sites*. Se, num caso extremo, existir a necessidade de comunicar com todos eles, o processo atrás referido teria que ser repetido pelo menos para cada *site*. Para solucionar esta situação, um novo caminho foi traçado.

Pretendia-se que, em vez de referenciar no MS SQL Server cada um dos *Web Services*, para os sistemas dos quais pretendemos obter a informação, apenas se referenciasse um único *Web Service*, o "Web Service Global". Este "Web Service Global" tem a responsabilidade de compilar e executar código Visual Basic .NET que recebe por parâmetro. O código que este recebe por parâmetro pode ser do mais diverso, desde simples operações em VB .NET que não seriam possíveis

na linguagem Transact-SQL, como por exemplo o *download* de um ficheiro da Internet utilizando credenciais de acesso, a complexas operações, tais como, referenciar e consumir outros *Web Services* de forma dinâmica, que é a solução pretendida neste projecto. Desta forma, com a implementação deste "Web Service Global", o problema de acesso por parte do MS SQL Server a dados de outros sistemas, via *Web Services*, ficou solucionado.

Em síntese, a comunicação entre o MS SQL Server e outros Sistemas de Informação recorrendo a diversos *Web Services*, foi conseguida através da implementação dos seguintes pontos:

- Criação de um "Web Service Global" capaz de compilar e executar código Visual Basic .NET que recebe como parâmetro. A linguagem Visual Basic .NET foi a adoptada neste projecto uma vez que é a mais conhecida por todos os programadores da Visabeira Digital;
- Associação do "Web Service Global" ao MS SQL Server. Este ponto é implementado através dos cinco ponto referidos no exemplo anterior, sobre como estabelecer uma ligação entre o MS SQL Server e um *Web Service* recorrendo a código CLR;
- Por último, a criação de uma função de nome "WebServiceInterface". É esta função que permite a passagem de parâmetros (código VB.NET) para o "Web Service Global" através da respectiva Assemblie. Uma vez que o MS SQL Server apenas aceita a transferência de tipos de dados serializáveis, o output do código executado no "Web Service Global" é do tipo *Dataset*. Neste projecto, o "Web Service Global" é utilizado para instanciar e invocar de forma dinâmica outros *Web Services* de outros Sistemas de Informação, eliminando assim a antiga necessidade de criar uma Assemblie por cada *Web Service*.

Na Ilustração 59 podemos analisar todo o processo de manipulação de dados entre o MS SQL Server e outros sistemas, recorrendo à utilização do "Web Service Global" acima descrito.

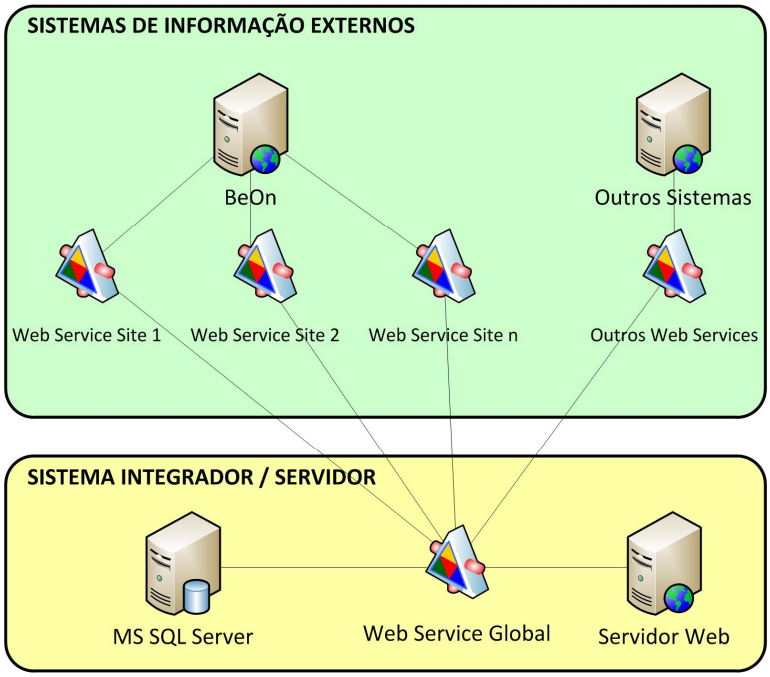

Ilustração 59 – Comunicação entre o MS SQL Server e as aplicações que disponibilizam Web Services.

O processo inicia-se no MS SQL Server que está enquadrado no Sistema Integrador/Servidor. Neste SGBD é executado o procedimento armazenado que contém todo o código VB .NET necessário para estabelecer o acesso ao *Web Service* do sistema de destino e executar o(s) respectivo(s) método(s). Tendo como exemplo o *download* de um ficheiro, este procedimento iria conter o código VB .NET que permite efectuar o respectivo *download*. Posteriormente invoca a função "WebServiceInterface" passando-lhe como parâmetro todo o código referido anteriormente. Nesta fase, a função "WebServiceInterface" é que permite a invocação e o envio do código VB .NET para o "Web Service Global", através de sua Assemblie criada anteriormente no MS SQL Server. Conforme podemos ver na Ilustração 59, o "Web Service Global" encontra-se publicado no Servidor Web pertencente ao Sistema Integrador/Servidor. Uma vez invocado o método de compilação e execução de código existente no "Web Service Global", este, de acordo com o código recebido, estabelece a ligação ao *Web Service* do Sistema de Informação Externo. Executa os respectivos métodos e devolve a informação de novo ao MS SQL Server para ser tratada por este.

A ilustração seguinte apresenta o código SQL para criar uma função capaz de fazer o *download* de um ficheiro, através da execução de código Microsoft Visual Basic .NET no "Web Service Global".

```
CREATE FUNCTION [dbo].[WEB_DownloadFile](
        @Url            VARCHAR(4000),
        @DomainAuth     VARCHAR(4000),
        @ProxyAuth      VARCHAR(4000)
) RETURNS XML
AS
BEGIN

DECLARE @Script         VARCHAR(MAX)

SET     @Script ='
<Root>
<Function>CodeCompileAndExecute</Function>
<Param1>WEbTools.DownloadFile</Param1>
<Param2>ghost</Param2>
<Param3>system.data.dll,system.xml.dll</Param3>
<Param4><![CDATA[
Imports System
Imports System.Xml.Serialization

Namespace WEbTools
Public Class DownloadFile

    Public Function ghost() As system.data.dataset
       Try
         dim down as new System.Net.WebClient
         dim ProxyAuth as string = "' + ISNULL(@ProxyAuth,'') + '"
         dim arrDomainAuth() as string = ("' + ISNULL(@DomainAuth,'') + '").Split(";")
         if arrDomainAuth.Length = 2
               down.Credentials = new System.Net.NetworkCredential(arrDomainAuth(0), arrDomainAuth(1))
         end if
           if arrDomainAuth.Length = 3
               down.Credentials = new System.Net.NetworkCredential(arrDomainAuth(0), arrDomainAuth(1), _
               arrDomainAuth(2))
         end if
         if not string.isnullorempty(ProxyAuth)
             dim arrProxyAuth() as string = ProxyAuth.Split(";")
             down.proxy = new System.Net.WebProxy(arrProxyAuth(0), true)
             if arrProxyAuth.Length = 3
                   down.proxy.Credentials = new System.Net.NetworkCredential(arrProxyAuth(1), arrProxyAuth(2))
             end if
               if arrProxyAuth.Length = 4
                   down.proxy.Credentials = new System.Net.NetworkCredential(arrProxyAuth(1), arrProxyAuth(2), _
                   arrProxyAuth(3))
             end if
         end if
         dim data() as byte = down.DownloadData("' + ISNULL(@Url,'') + '")
         dim ds as new system.data.dataset
         ds.tables.add("File").columns.add("FileData",System.Type.GetType("System.Byte[]") )
         ds.tables(0).rows.add(data)
         return ds
        Catch ex2 as system.Exception
            return ghostex(ex2)
        End Try
        me.Finalize()
    End Function

    Public Function ghostex(ex as System.Exception) As System.Data.DataSet
        Dim ds as new data.dataset("Exceptions")
        Dim dt As System.Data.DataTable = ds.Tables.Add("Exception")
        dt.Columns.Add("Ex", GetType(String))
        dt.Columns.Add("INNER", GetType(String))
        Dim dr AS system.data.datarow = dt.rows.add()
        dr("Ex") = ex.GetType().InvokeMember("ToString", (System.Reflection.BindingFlags.Default Or
        System.Reflection.BindingFlags.InvokeMethod), Nothing, ex, Nothing)
        if ex.GetType().tostring().contains("SoapException")
           dt.Columns.Add("Detail", GetType(String))
           dr("Detail") = ex.GetType().GetProperty("Detail").getValue(ex,nothing).innertext
        end if
        IF not(ex.InnerException is nothing)
           dr("INNER") = ex.InnerException.tostring()
        END if
        return ds
     End Function
End Class

End Namespace
    ]]></Param4>
</Root>';

DECLARE @returnXML XML
SELECT @returnXML = dbo.[WebServiceInterface](@Script)

RETURN @returnXML
END
```

Ilustração 60 – Função SQL com código VB .NET para efectuar o *download* de um ficheiro.

Rapidamente, analisando a Ilustração 60, apercebemo-nos que a função SQL apresentada recebe como parâmetros o URL do ficheiro de que será feito *download*, e a autenticação de domínio e

proxy, se esta for necessária. Estes parâmetros permitem a fácil reutilização da mesma função em diversas situações. Analisando o código da função verificamos que a variável "Script" contém uma estrutura XML. Relembrando o que foi dito anteriormente, o MS SQL Server apenas permite a passagem de tipos de dados serializáveis para as Assemblies. Desta forma, o tipo de dados XML, para além de ser serializável, permite o rápido acesso aos dados armazenados nos seus nós. Esta estrutura foi organizada em cinco nós principais, que foram identificados como os nós indispensáveis para o correcto funcionamento implementado no "Web Service Global", são eles:

- Function – que corresponde à função a ser invocada no "Web Service Global". No exemplo apresentado o código irá ser executado na função "CodeCompileAndExecute", que é a função responsável por compilar e executar código em tempo real. O objectivo deste nó é a possibilidade de executar o código VB .NET enviado, em outras funções com comportamentos diferentes. Por exemplo, durante a fase de melhoramento do "Web Service Global", pode ser compilado código noutra função sem influenciar o actual funcionamento do sistema.
- Param1 – define o *namespace* e a classe na qual está declarado o código VB .NET. O *namespace* e a classe são indispensáveis para se poder instanciar correctamente a função declarada no "Param2".
- Param2 – indica o nome da função a executar. Esta função deverá estar declarada no código (Param4) e terá que estar localizada no *namespace* e classe definida no "Param1". É esta função que irá desencadear a execução do código após este ter sido compilado. Analisando a Ilustração 60, verificamos que este parâmetro tem o valor "*ghost*", quer isto dizer que o "Web Service Global", após compilar o código irá invocar a função com o nome "*ghost*".
- Param3 – responsável pela inclusão de referências a outras DLL's, que caso participem no código VB .NET terão que obrigatoriamente estar presentes no processo de compilação de código e no servidor *Web* onde está alojado o "Web Service Global".
- Param4 – corresponde ao conteúdo/código que será compilado e executado pelo "Web Service Global".

Desta forma, o "Web Service Global" apenas funcionará se o código enviado seguir a estrutura XML apresentado na Ilustração 60.

Ainda no contexto da função que permite o *download* de um ficheiro através de código Microsoft Visual Basic .NET, a ilustração abaixo apresentada expõe um exemplo da invocação dessa mesma função.

```
SELECT dbo.WEB_DownloadFile(
            'http://google.pt/images/srpr/logo3w.png',  -- url do ficheiro
            '[illegible]',                              -- autenticação do domínio
            '[illegible]' -- autenticação do proxy
            )
```

Ilustração 61 – Exemplo de invocação da função de *download* de um ficheiro, por SQL.

Reutilizando este princípio foram criadas outras funções. Uma delas tem o nome de "GetListItems". Esta função contém código Visual Basic .NET responsável por estabelecer uma ligação a

um *site* do SharePoint, através do *Web Service* "lists.asmx", e invocar o método "GetListItems" que permite a consulta de dados de uma determinada lista passada por parâmetro. Com o objectivo de permitir a reutilização desta função para aceder a outras listas e outros *sites* do SharePoint, a função foi criada tendo em conta os parâmetros de entrada identificados na Ilustração 62. Se analisarmos o código apresentado nesta ilustração, verificamos que na função "ghost" é instanciado um objecto do tipo "Lists". Este objecto é definido por uma classe "Lists" declarada no próprio código VB .NET. De igual modo, como podemos ver, o código representado na Ilustração 62 contém todas as referências necessárias para que este seja compilado e executado sem qualquer problema.

```
CREATE FUNCTION [dbo].[GetListItems](
        @Url                VARCHAR(4000),
        @DomainAuth         VARCHAR(4000),
        @ProxyAuth          VARCHAR(4000),
        @ListName           VARCHAR(4000),
        @ViewName           VARCHAR(4000),
        @CamposOWSS         VARCHAR(4000),
        @QueryXMLWSS        VARCHAR(max),
        @RowLimitWSS        int,
        @QueryOptionsWSS    VARCHAR(4000) = ''
) RETURNS XML
AS
BEGIN

DECLARE @Script                 VARCHAR(MAX)

SET     @Script = '
<Root>
<Function>CodeCompileAndExecute</Function>
<Param1>WSSTools.ListsProxy</Param1>
<Param2>ghost</Param2>
<Param3>system.data.dll,System.Web.Services.dll,system.xml.dll</Param3>
<Param4><![CDATA[

Imports System
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.Xml.Serialization

Namespace WSSTools
Public Class ListsProxy

    Public Function ghost() As system.data.dataset
        Try
          dim xDoc as new Xml.XmlDocument()
          dim xQuery as xml.xmlelement = xDoc.CreateElement("Query")
          xQuery.InnerXML = "' + ISNULL(@QueryXMLWSS,'') + '"

          dim xFields as xml.xmlelement = xDoc.CreateElement("ViewFields")
          for each field as string in ((new string("' + ISNULL(@CamposOWSS,'') + '")).split(","))
              dim xField as XML.XMLElement = xDoc.CreateElement("FieldRef")
              xField.SetAttribute("Name",field)
              xFields.AppendChild(xField)
          NEXT

          dim xQOption as xml.xmlelement = xDoc.CreateElement("QueryOptions")
          xQOption.InnerXML = "<IncludeAttachmentUrls>True</IncludeAttachmentUrls><IncludeMandatoryColumns>TRUE" & _
                              "</IncludeMandatoryColumns>'+ ISNULL(@QueryOptionsWSS,'') + '"

          dim list as new lists("' + ISNULL(@Url,'') + '","' + ISNULL(@DomainAuth,'') + '", "' + ISNULL(@ProxyAuth,'') + '" )
          dim DS as new system.data.dataset
          ds.readxml(new system.io.stringreader(list.GetListItems("' + ISNULL(@ListName,'') + '", "' + ISNULL(@ViewName,'') + '", _
                    xQuery,xFields,' + ISNULL(cast(@RowLimitWSS as varchar(20)),'9999999') + ',xQOption,nothing).outerxml))
          return ds
        Catch ex2 as system.Exception
           return ghostex(ex2)
        End Try
        me.Finalize()
    End Function

    Public Function ghostex(ex as System.Exception) As System.Data.DataSet
     End Function
End Class


<System.Diagnostics.DebuggerStepThroughAttribute(),  _
 System.ComponentModel.DesignerCategoryAttribute("code"),  _
 System.Web.Services.WebServiceBindingAttribute(Name:="ListsSoap", [Namespace]:="http://schemas.microsoft.com/sharepoint/soap/")>  _
Partial Public Class Lists
    Inherits System.Web.Services.Protocols.SoapHttpClientProtocol

    Public Sub New(ByVal SiteURL As String, Byval DomainAuth AS string,ByVal ProxyAuth as string)
       MyBase.New
       If Not SiteURL.EndsWith("/") Then
           SiteURL &= "/"
        End If
        Me.Url = SiteURL & "_vti_bin/lists.asmx"
        dim arrDomainAuth() as string = DomainAuth.Split(";")
        if arrDomainAuth.Length = 2
               me.Credentials = new System.Net.NetworkCredential(arrDomainAuth(0), arrDomainAuth(1))
         end if
           if arrDomainAuth.Length = 3
                me.Credentials = new System.Net.NetworkCredential(arrDomainAuth(0), arrDomainAuth(1), arrDomainAuth(2))
         end if
         if not string.isnullorempty(ProxyAuth)
             dim arrProxyAuth() as string = ProxyAuth.Split(";")
             me.proxy = new System.Net.WebProxy(arrProxyAuth(0), true)
             if arrProxyAuth.Length = 3
                   me.proxy.Credentials = new System.Net.NetworkCredential(arrProxyAuth(1), arrProxyAuth(2))
             end if
               if arrProxyAuth.Length = 4
                   me.proxy.Credentials = new System.Net.NetworkCredential(arrProxyAuth(1), arrProxyAuth(2), arrProxyAuth(3))
             end if
         end if
    End Sub

    <System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://schemas.microsoft.com/sharepoint/soap/GetListItems", _
    RequestNamespace:="http://schemas.microsoft.com/sharepoint/soap/", ResponseNamespace:="http://schemas.microsoft.com/sharepoint/soap/", _
    Use:=System.Web.Services.Description.SoapBindingUse.Literal, ParameterStyle:=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)> _
    Public Function GetListItems(ByVal listName As String, ByVal viewName As String, ByVal query As System.Xml.XmlNode, _
                    ByVal viewFields As System.Xml.XmlNode, ByVal rowLimit As String, ByVal queryOptions As System.Xml.XmlNode, _
                    ByVal webID As String) As System.Xml.XmlNode
        Dim results() As Object = Me.Invoke("GetListItems", New Object() {listName, viewName, query, viewFields, rowLimit, queryOptions, webID})
        Return CType(results(0),System.Xml.XmlNode)
    End Function
End Class
End Namespace
    ]]></Param4>
</Root>';

DECLARE @retXML XML
SELECT @retXML = dbo.[WebServiceInterface](@Script)
RETURN @retXML
END
```

Ilustração 62 – Criação da função SQL que permite o acesso a dados do SharePoint através do método GetListItems.

Com estes parâmetros o acesso a dados de listas do SharePoint é simplificado como podemos comprovar na ilustração seguinte.

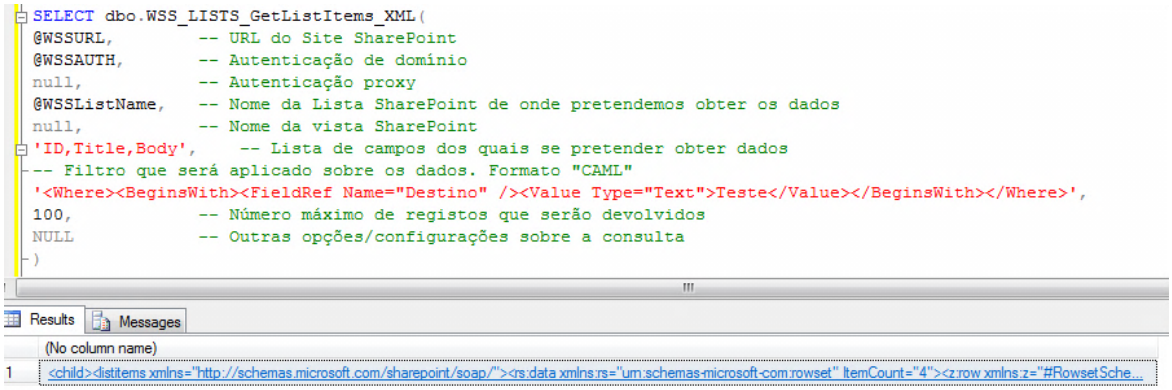

Ilustração 63 – Obtenção de dados de uma lista em SharePoint através da função SQL.

Analisando a Ilustração 63, o output devolvido é um valor no formato XML. Para simplificar a manipulação desta informação, são utilizados métodos XQuery e XPath que permitem a visualização e o tratamento desta informação no formato de tabela SQL.

De igual modo outras funções foram ser criadas, sob o mesmo princípio mas com diferentes objectivos, tais como a inserção e actualização de dados em listas do SharePoint e de outros Sistemas de Informação Externos.

Todas as funções SQL anteriormente apresentadas são utilizadas no Sistema Integrador/Servidor, quer no processo de consulta de dados dos Sistemas de Informação Externos (BeOn), quer no processo de integração de dados para estes mesmos Sistemas de Informação.

A integração das Guias de Transporte, objecto que iniciou todo o processo de investigação deste projecto, é efectuada com o módulo MM do SAP. Para satisfazer este passo, o Sistema Integrador/Servidor analisa o XML que contém toda a informação da Guia de Transporte, com o intuito de determinar o tipo de movimento e a transacção que este terá que gerar no sistema SAP. Após determinar estes dois tipos de informação, o Sistema Integrador/Servidor invoca o respectivo procedimento armazenado que será o responsável por estabelecer a ligação ao sistema SAP, gerar a BAPI que permite a execução da transacção definida para o tipo de movimento em questão, e, por fim, proceder ao lançamento da informação. Para estabelecer a ligação ao sistema SAP, o processo é muito semelhante ao acesso via *Web Services* descrito anteriormente. Neste caso, é utilizado o protocolo HTTP em conjunto com as BAPIs referidas também no ponto 4.1.3 deste capítulo. Mais uma vez o acesso ao sistema SAP é efectuado recorrendo ao "Web Service Global". Este *Web Service* estabelece a comunicação com o sistema SAP e submete o pedido de acesso formatado em XML referente à BAPI pretendida. Recorrendo à Ilustração 64 podemos analisar este processo através dos pontos 1 e 2.

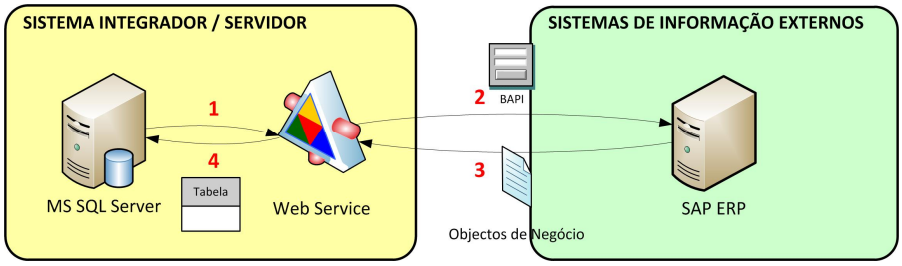

Ilustração 64 – Acesso ao Sistemas de Informação Externos SAP.

O ponto 3 da ilustração acima apresentada representa o fluxo de obtenção de dados do SAP após execução dos comandos enviados no formato de BAPI. O SAP devolve a informação da operação (dados, mensagens de erro ou sucesso, etc.) através de Objectos de Negócio também no formato XML. Para facilitar o tratamento desta informação, foi criada uma função no MS SQL Server, de nome "RenderXML", responsável por converter os Objectos de Negócio no formato XML em tabelas de carácter temporário. A conversão ocorre já na Base de Dados MS SQL Server após ultrapassado o ponto 4 da Ilustração 64. É desta forma que o Sistema Integrador/Servidor consegue comunicar com o Sistema de Informação SAP. O processo agora descrito é válido, não só para a obtenção de informação de SAP, mas também para o processo de integração, onde são devolvidos Objectos de Negócio com a informação relativa à integração (número do documento gerado, mensagens de erros, etc.). Esta informação é analisada e posteriormente armazenada no campo "IntegrationResult" da entidade "CacheLogos", permitindo a futura correcção do registo e a sua reintegração, se for o caso.

4.4.1.2 Activação do mecanismo de actualização e integração de dados

Em relação ao modo como o mecanismo de actualização e integração de dados é activado, pretende-se a implementação de uma ferramenta capaz de invocar, em determinados intervalos de tempo, os procedimentos armazenados responsáveis pela actualização/integração de dados referidos na subsecção anterior. Pretende-se ainda que cada conjunto de dados (Guias de Transporte, Firmas, Centros Logísticos, Materiais, etc.) seja actualizado/integrado em intervalos de tempo distintos, uma vez que, em alguns casos, os dados não variam durante algum tempo. Assim, tendo estes objectivos em foco, identificaram-se as seguintes soluções suportadas pelo Sistema Integrador/Servidor, são elas: Serviços do Windows, Aplicações de Consola agendada com tarefas do Windows e *Jobs* SQL (SQL Server Agent Jobs) [MAGALHÃES, 2008]. Todas as soluções apresentadas permitem a implementação de múltiplos agendamentos com periodicidades distintas. Contudo, de entre as soluções de agendamento identificadas, e uma vez que toda a comunicação com os Sistemas de Informação Externos é efectuada através do MS SQL Server, a solução adoptada foi a criação de *Jobs* SQL. Esta ferramenta está integrada no próprio MS SQL Server, instalado no Sistema Integrador/Servidor, não necessitando de qualquer tipo de desenvolvimento, ao contrários das restantes soluções apresentadas. Os *Jobs* SQL foram assim agendados e configurados com blocos de código Transact-SQL responsáveis por invocar procedimentos armazenados, de forma a satisfazer todas as necessidades impostas ao mecanismo de actualização e integração de dados. No caso da actualização de dados, os procedimentos armazenados, contendo código semelhante ao da Ilustração 63, deverão aceder a dados dos Sistemas de Informação Externos e proceder à respectiva actualização da Base de Dados local do Sistema Integrador/Servidor. Para as situ-

ações de integração de dados, serão invocados procedimentos armazenados responsáveis por estabelecer uma ligação a um Sistema de Informação Externo e integrar os dados que lhe dizem respeito. No exemplo das Guias de Transporte, o procedimento armazenado estabelece uma ligação ao SAP ERP e executa as transacções necessárias ao registo da informação relativa a cada Guia de Transporte. Após integração de cada Guia de Transporte, o procedimento armazenado actualiza o seu registo, na entidade "CacheLogos" existente na Base de Dados local ao Sistema Integrador/Servidor, com o respectivo resultado (output) devolvido pelo sistema SAP.

4.4.2 Acesso directo a dados dos Sistemas de Informação Externos

O acesso directo a dados dos Sistemas de Informação Externos ocorre quando este é solicitado por funcionalidades *online* disponíveis no Sistema Cliente.

Uma vez que os Sistemas Clientes estão fora da rede interna era necessário implementar uma interface que permitisse o seu acesso ao Sistema Integrador/Servidor. De acordo com a arquitectura proposta o interface implementado foi um *Web Service* para o qual o acesso está devidamente configurado no Sistema Cliente, através da entidade "ConfigCache".

Este *Web Service* implementa um método específico para o acesso directo a dados, denominado de "FunctionalitiesData", o qual inclui três parâmetros:

- ConnectionString – responsável por receber os dados para estabelecer uma ligação à Base de Dados do Sistema Integrador/Servidor;
- Functionality – indica a funcionalidade que está a solicitar os dados;
- Parameters – permite a passagem de outros parâmetros utilizados na obtenção de dados.

Assim sendo, uma determinada funcionalidade do Sistema Cliente, a pedido do utilizador, invoca este *Web Service* com os respectivos parâmetros, ponto 1 e 2 da Ilustração 65.

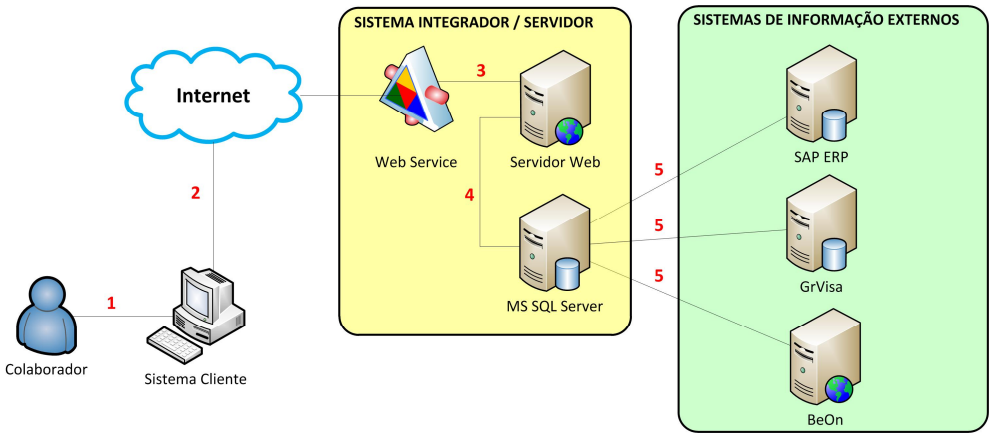

Ilustração 65 – Esquema de comunicação de dados *online*.

Por sua vez, o *Web Service* utiliza o valor do parâmetro "ConnecionString" para estabelecer uma ligação à Base de Dados do Sistema Integrador/Servidor (ponto 3 da Ilustração 65) e invoca um procedimento armazenado, de nome "WebService_FunctionalitiesData", passando-lhe os parâmetros "Functionality" e "Parameters" (ponto 4 da ilustração referia anteriormente). À seme-

lhança do que foi explicado na secção anterior (por exemplo a invocação da função "GetListItems"), existem diversos procedimentos armazenados que implementam a lógica de negócio e os métodos de acesso aos dados dos diversos Sistemas de Informação Externos (passo 5 da Ilustração 65). Assim sendo, cabe ao procedimento armazenado "WebService_FunctionalitiesData" analisar o parâmetro "Functionality" e, mediante o seu valor, invocar o respectivo procedimento armazenado, responsável por devolver os dados solicitados.

4.5 Instalação e configuração

O processo de instalação e configuração é dos mais importantes neste sistema para que tudo funcione bem. A instalação em larga escala face ao número de máquinas dos utilizadores (Sistema Cliente) requer alguma atenção. Para tal, optou-se por utilizar o método de instalação *ClickOnce* [ClickOnce, 2011] disponível no Microsoft Visual Studio 2010. Este modo de instalação é *online*, isto é, permite a publicação do ficheiro de instalação da aplicação informática num servidor *Web* e a distribuição do respectivo URL. Com este URL o utilizador pode efectuar a instalação da aplicação informática através de um único clique conforme apresenta a Ilustração 66.

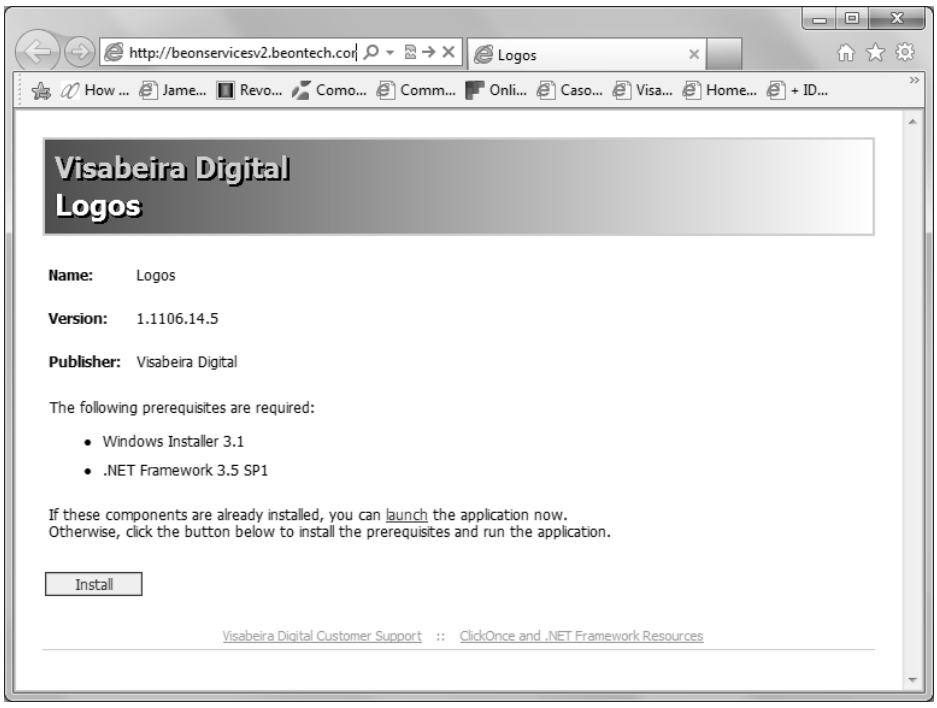

Ilustração 66 – Instalação do Sistema Cliente por ClickOnce.

O método de instalação *ClickOnce* permite ainda a instalação de outros componentes e/ou pré-requisitos da aplicação. O pré-requisito para a instalação da aplicação é a Framework .NET 3.5. Após validação deste pré-requisito o processo de instalação segue para a instalação do Microsoft SQL Server 2008 R2 Express, embutido no pacote de instalação [EmbeddingSQLServer, 2011], e configuração da respectiva Base de Dados. O passo seguinte é a instalação do mecanismo de sincronização. Este, como foi anteriormente referido, actua sob a forma de um Windows Service que está sempre activo e em execução. Para finalizar todo o processo de instalação do Sistema Cliente,

segue a instalação da aplicação informática. Para um fácil acesso à aplicação, por norma, é criado um ícone no ambiente de trabalho e outro no menu iniciar.

Outra vantagem deste método de instalação é o fácil lançamento de novas actualizações. Uma vez que a aplicação está publicada, quando é feita uma nova publicação todos os Sistemas Clientes que iniciarem no modo *online* receberão a notificação de que uma nova versão da aplicação se encontra disponível. Para efectuar a actualização da aplicação, o utilizador só tem que confirmar a operação. Após confirmação é feito o *download* e a instalação da nova versão de forma automática e transparente para o utilizador.

4.6 Ferramentas e linguagens de programação utilizadas

Uma vez que o desenvolvimento de *software* efectuado pela Visabeira Digital é baseado em ferramentas e produtos Microsoft, a escolha das ferramentas adoptadas para o desenvolvimento e implementação dos componentes objecto da arquitectura proposta, foi simplificada.

Como ferramenta de programação adoptou-se o IDE Microsoft Visual Studio 2010 em conjunto com o Team Foundation Server (servidor de ficheiros com controlo de versões). Estas duas ferramentas permitem, de uma forma fácil, a criação e gestão de projectos orientada para o desenvolvimento colaborativo de *software*.

Como ferramenta de auxílio para operações sobre a Base de Dados, foi adoptado o SQL Management Studio. O SQL Server Management Studio é a ferramenta que possibilita a gestão e configuração de todas as funcionalidades do SQL Server, simples de usar e extremamente útil. Com ele vem um conjunto de funcionalidades para facilitar a sua manutenção, apresentando relatórios de análise capazes de alertar o programador para situações que possam influenciar o seu desempenho.

Por outro lado, tendo em conta que a linguagem de programação mais abrangente entre os colaboradores da Visabeira Digital é Visual Basic .NET, esta foi a linguagem adoptada no desenvolvimento de todo o *software* relativo a este projecto, já que, no futuro, serão desenvolvidos muitos outros componentes e módulos de *software* por outros elementos da equipa de desenvolvimento de *software*, módulos estes que serão integrados nesta arquitectura.

De igual forma, foi utilizada a linguagem de programação ASPx.NET e VB.NET para a criação de canais de comunicação (*Web Services*) inexistentes.

4.7 Problemas encontrados

Durante a fase de implementação surgiram algumas dificuldades que foram ultrapassadas mediante testes, caminhos abandonados e pesquisas efectuadas a outras soluções alternativas. A presente secção identifica alguns dos principais problemas e, de forma sucinta, explica de que forma é que estes foram solucionados.

Uma vez que a instalação e configuração dos Sistemas Clientes é feita por uma equipa de suporte, com o arranque da fase de testes deste projecto, não havia a noção de quantas e quais eram as máquinas que tinham o Sistema Cliente e qual a versão que cada uma tinha instalada. Para obter esta informação, a equipa de suporte teria que enviar um *e-mail* para cada colaborador a solicitar esta informação. Nas situações em que os colaboradores não sejam capaz de identificar a informação solicitada, a equipa de suporte teria que estabelecer uma ligação remota a cada máquina. A solução para esta situação passou por enviar para os Sistemas Clientes, através da entidade "CacheOperations" (segundo o processo descrito na secção 4.2), blocos de código SQL que permitiram obter a informação pretendida (nome da máquina, endereço IP, versão da aplicação, entre outros). Após a execução destes blocos de código SQL pela aplicação base, toda a informação foi registada na entidade "CacheLogos" como sendo de uma funcionalidade de nome "WHOIS". Em poucos minutos os Sistemas Clientes começaram a enviar a sua informação para o Sistema Integrador/Servidor e rapidamente foi detectado um outro problema.

Ao confrontar a informação recebida dos Sistemas Clientes com as listagens das máquinas instaladas pela equipa de suporte, verificou-se que algumas delas não estavam a comunicar. Face à distância a que as máquinas se encontravam, efectuou-se uma ligação remota para averiguar e detectar a causa para o problema identificado. Rapidamente se detectou que o mecanismo de actualização e sincronização de dados existente no Sistema Cliente, sob a forma de *Windows Service*, estava parado. Consultaram-se os eventos do Windows e *logs* do sistema e chegou-se à conclusão que a causa por este estar parado, surgiu devido a uma excepção de "*OutOfMemory*". Vários testes foram feitos ao *Windows Service* verificando-se que, embora a programação seguisse as boas práticas e libertasse todos os recursos envolvidos, cada vez que o *Windows Service* era executado, a memória utilizada por este aumentava e nunca reduzia, nem mesmo após conclusão da sua execução.

Após algumas pesquisas encontraram-se referencias a problemas semelhantes que remetiam a causa do problema "*OutOfMemory*" para uma falha de memória (*memory leak*) do *Windows Service*. A solução imediata passou pela criação de um *script* para reiniciar o *Windows Service* quando a memória consumida por este atingisse o valor de 100 *megabytes*. Mais tarde o problema viria a ser identificado e corrigido, procedendo-se assim à eliminação do *script* anteriormente referido. O problema causador das falhas de memória estava relacionado como os recursos consumidos por algumas variáveis, utilizadas pelo mecanismo de actualização e integração de dados, que, devido ao seu *scope* e à sua implementação multitarefa, nunca eram libertados.

Durante a implementação do processo de integração modular das funcionalidades, surgiu uma outra dificuldade. Esta estava relacionada com as dependências que cada funcionalidade apresen-

tava a outros componentes externos, nomeadamente os componentes de produtividade DevExpress e o relatório de impressão da Guia de Transporte. Neste caso, foram adoptadas duas soluções distintas.

Relativamente às referências para os componentes de produtividade DevExpress, e uma vez que estes eram utilizados por todos os módulos de *software* (funcionalidades), optou-se por adicionar todas as referências dos componentes DevExpress na aplicação base. Esta solução permitiu reduzir o tamanho dos módulos de *software*, assim como centralizar os componentes de produtividade num único ponto, a aplicação base. Se por acaso estes sofrerem uma actualização, automaticamente todas as funcionalidades passarão a utilizar a nova versão.

Já o relatório de impressão da Guia de Transporte é específico à funcionalidade de emissão de Guias de Transporte. Assim sendo, a DLL responsável por este relatório foi incorporada na funcionalidade de emissão de Guias de Transporte como um recurso incorporado (*Embedded Resource*). Ao gerar a DLL referente à funcionalidade o relatório ficará embutido nesta, sendo posteriormente a funcionalidade a responsável por carregar e apresentar o relatório quando este for solicitado.

Em síntese, todos os componentes apresentados nesta arquitectura foram implementados com sucesso. No Sistema Cliente foi implementada a aplicação base, uma Base de Dados e o Mecanismo de Sincronização de Dados cujo funcionamento já foi apresentado na subsecção 3.1.1. Já no Sistema Integrador/Servidor foi também implementada uma Base de Dados, o Mecanismo de Actualização e Integração de dados e um mecanismo que permite o acesso directo a dados dos Sistemas de Informação Externos, introduzidos na subsecção 3.1.2.

Capítulo 5

Conclusões e Trabalho Futuro

O capítulo 5, último deste documento, contém as considerações finais mais importantes a serem recordadas. Descreve ainda as principais conclusões, contribuições e orientações para possíveis trabalhos futuros.

5.1 Avaliação Crítica

Este projecto teve a sua génese, no Grupo Visabeira, durante a substituição do seu ERP, o GrVisa, pelo SAP. Durante este processo surgiram algumas necessidades de suporte à operação, sendo que uma delas foi identificada por uma empresa de telecomunicações constituinte do Grupo, a Viatel.

A Viatel, alertou para a necessidade de adquirir um sistema *offline* que permitisse a emissão de Guias de Transporte, mesmo aquando falhas de ligação entre os seus armazéns e o servidor central, para a qual o SAP não era capaz de suportar.

Sendo a Visabeira Digital a empresa responsável pelo *software* do Grupo Visabeira, foi incumbida de analisar a possibilidade de implementar não só a funcionalidade de emissão de Guias de Transporte no modo *offline*, mas também a possibilidade de estruturar uma arquitectura de interligação global que garantisse a expansão do sistema a novas funcionalidades e a outras empresas. O sistema resultante desta análise deveria assegurar ainda a integração dos dados com o novo ERP.

O presente trabalho iniciou-se com um estudo aprofundado sobre os sistemas *Web-Based* e a arquitectura cliente/servidor de forma a aproveitar as suas vantagens para o desenvolvimento de uma nova arquitectura. Esta nova arquitectura foi estruturada e subdividida em três sistemas principais: o Sistema Cliente, o Sistema Integrador/Servidor e os Sistemas de Informação Externos.

O Sistema Cliente representa um qualquer dispositivo terminal a que os utilizadores recorrem para efectuar operações, por exemplo a emissão de Guias de Transporte. Um dos requisitos da Viatel era que a funcionalidade de emissão de Guias de Transporte pudesse operar sem a necessi-

dade de estabelecer uma ligação ao sistema central. Para superar este requisito o Sistema Cliente implementa um funcionamento dual, permitindo que as funcionalidades disponibilizadas por este possam aceder a dados em tempo real (modo *online*), operar recorrendo exclusivamente aos dados armazenados localmente, sem estabelecer uma ligação ao servidor, (modo *offline*) e ainda, estabelecer um consenso entre ambos os modos de funcionamento.

No modo *offline*, cada funcionalidade, caso necessite, recorre a tabelas de apoio disponíveis na Base de Dados local do Sistema Cliente. Estas tabelas de apoio (introduzidas na secção 4.2) são devidamente sincronizadas pelo mecanismo de sincronização de dados apresentado na subsecção 4.3.2. Toda a informação gerada pela funcionalidade é devidamente armazenada na entidade "CacheLogos" no formato de XML. A estrutura de dados no formato XML é definida por cada funcionalidade e será a responsável por garantir todos os seus requisitos de informação. Esta informação é posteriormente sincronizada para o Sistema Integrador/Servidor aquando o Sistema Cliente passa para o modo *online*. É graças ao armazenamento da informação no formato XML, que o sistema permite facilmente a sua expansão a novas funcionalidades, não sendo necessária qualquer alteração a nível da sua estrutura.

Outro requisito da Viatel era a recolha da assinatura digital do técnico pela funcionalidade de emissão de Guias de Transporte. Para garantir esta opção foi implementado um pequeno módulo de *software* que recolhe a assinatura através de um modo gráfico, no qual é usada uma caneta digital para introduzir a assinatura. A informação proveniente deste módulo de *software* é convertida e armazenada no próprio XML da Guia de Transporte.

Os acessos às funcionalidades são atribuídos por utilizador e garantidos pelo modelo lógico de dados apresentado na secção 4.2. Este modelo lógico de dados permite ainda a redução dos dados que serão sincronizados, quer isto dizer que para um determinado Sistema Cliente apenas serão sincronizadas as funcionalidades a que o utilizador desse Sistema Cliente possui acesso.

O canal de comunicação que permite o envio de mensagens para os utilizadores, requisito imposto pela Viatel, foi implementado pela entidade "CacheOperations". Os dados desta entidade apenas são sincronizados num único sentido (do Sistema Integrador/Servidor para o Sistema Cliente). No Sistema Cliente as funcionalidades que operam sobre esta entidade são as responsáveis por verificar a disponibilidade de novos registos, analisando posteriormente o seu conteúdo e prosseguindo com o seu funcionamento. Sobre este processo, para além da funcionalidade que apresenta as notificações/mensagens aos utilizadores, foi também implementada uma funcionalidade que executa código SQL enviado através desta entidade, directamente na Base de Dados local. O objectivo desta funcionalidade é enviar e executar blocos de código SQL em todos ou em determinados Sistemas Clientes, permitindo assim efectuar alterações sobre a Base de Dados e sobre os seus registos.

Um dos objectivos propostos pela Visabeira Digital era a expansão do Sistema Cliente a outras empresas e áreas de negócio. Desta forma, o Sistema Cliente teria que suportar e disponibilizar um conjunto de funcionalidades completamente distintas. Com o objectivo de facilitar o modo como a expansão do Sistema Cliente seria feita, implementou-se uma estrutura modular de *software*. Assim, o desenvolvimento de novas funcionalidades é simplificado e traduz-se na implementação de

módulos de *software* no formato de *user controls*. Estes módulos de *software,* após compilados através do IDE Microsoft Visual Studio 2010, geram um ficheiro do tipo DLL que é posteriormente armazenado na Base de Dados e sincronizado para os respectivos Sistemas Clientes. Uma vez no Sistema Cliente, e após invocação por parte do utilizador, o módulo de *software* é carregado e apresentado ao utilizador, através da aplicação informática. Todo este processo garante o desenvolvimento modular de *software* e a fácil expansão do número de funcionalidades disponíveis no Sistema Cliente.

Ainda no âmbito do desenvolvimento de *software* foram criados dois estágios: um de Desenvolvimento/Qualidade, onde prevalecem e são testadas as funcionalidades provenientes do desenvolvimento de *software*, e outro de Produção, onde estão publicadas as funcionalidades e os dados reais de acesso aos utilizadores finais.

Uma vez que este sistema permite a utilização de funcionalidade no modo *offline*, era também um requisito que a aplicação implementasse um mecanismo de segurança que permitisse o bloqueio de acesso às suas funcionalidades. Assim, o mecanismo implementado bloqueia a aplicação caso esta esteja a operar no modo *offline* durante um período de tempo superior ao permitido. Este bloqueio é efectuado tendo em conta a data do último acesso da aplicação ao Sistema Integrador/Servidor, como é explicado na subsecção 4.3.1.

Com a expansão deste projecto a todas as empresas do Grupo Visabeira, incluindo multinacionais, a Visabeira Digital identificou devidamente a necessidade de o Sistema Cliente conseguir apresentar as suas funcionalidades em diversas línguas. Não pretendendo recorrer a nenhum motor de traduções *online*, devido ao funcionamento dual do Sistema Cliente, foi criada uma tabela de apoio responsável por armazenar as diferentes traduções para cada descrição/frase apresentada pelas funcionalidades. O processo de carregamento da língua pretendida ocorre no Sistema Cliente, após a escolha do idioma no ecrã de *login*. Efectuado o *login*, cada vez que uma funcionalidade é apresentada ao utilizador, esta verifica primeiro o idioma seleccionado e recorre depois à tabela de apoio para traduzir o seu conteúdo.

Já o Sistema Integrador/Servidor foi criado com o objectivo de centralizar o acesso aos Sistemas de Informação Externos num único ponto. Desta forma, toda a integração de dados entre os diferentes Sistemas de Informação é validada e executada por este sistema, quer seja esta integração efectuada no modo síncrono (*on-time*) ou no modo assíncrono (através de rotinas de integração). É este Sistema Integrador/Servidor que é o responsável por integrar, entre outros, as Guias de Transporte provenientes dos Sistemas Clientes com o respectivo módulo do SAP ERP, garantindo assim o requisito imposto pela Viatel.

Também o número de licenças necessárias para o SAP foi reduzido, uma vez que o Sistema Integrador/Servidor utiliza sempre o mesmo utilizador durante os vários acessos que estabelece ao sistema. Os restantes utilizadores (técnicos, fiéis de armazém, etc.) não chegam sequer a aceder ao sistema SAP, estes utilizam os dados disponíveis no Sistema Cliente, ou aquando estabelecido o acesso a dados *online* utilizam, mais uma vez, o utilizador do Sistema Integrador/Servidor.

Em relação à redução de custos relativos aos contratos de fornecimento de *links* dedicados entre o sistema central e os armazéns, foram efectuados alguns testes nas máquinas dos colaboradores da Visabeira Digital. Nestes testes substituíram-se as ligações de rede por ligação 3G. Após esta substituição, verificou-se que o Sistema Cliente conseguia comunicar e sincronizar dados sem problema. Desta forma, e embora ainda não tenham sido elaborados testes em produtivo, conclui-se que a substituição dos *links* dedicados por placas 3G é possível graças ao comportamento dual que o sistema permite (modo *online* ou *offline*).

A ilustração apresentada de seguida identifica de forma sucinta as principais características da arquitectura proposta.

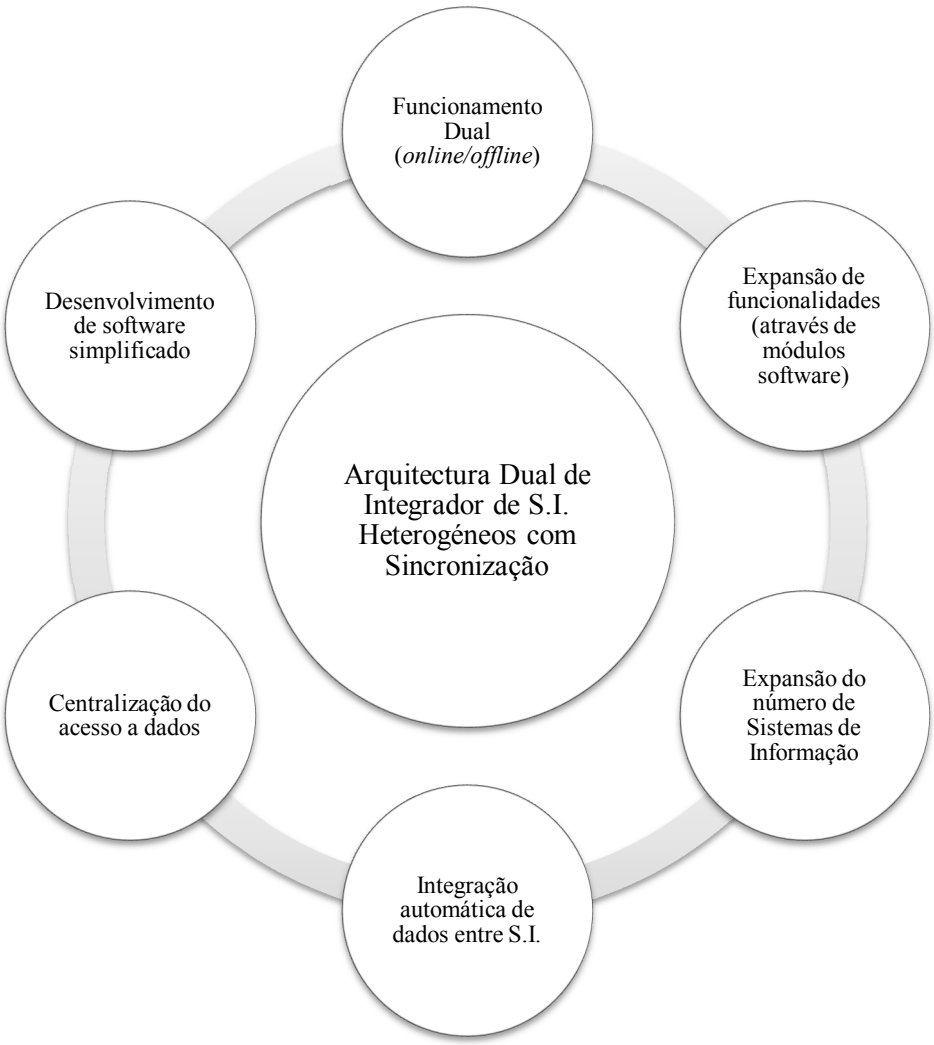

Ilustração 67 – Principais características da arquitectura proposta.

Constata-se que todos os objectivos propostos inicialmente nesta dissertação foram totalmente atingidos e superados com sucesso. O projecto implementado demonstrou-se do agrado da Viatel e da Visabeira Digital. Por um lado, a Viatel ficou com um Sistema de Informação desenvolvido de acordo com as suas necessidades. Este sistema não só possui mecanismos que permitem acelerar os seus processos de negócio, como garante a integração dos dados com o SAP ERP. Os utilizadores da Viatel também demonstraram o seu agrado pela aplicação informática, não só a nível do *design* que as funcionalidades apresentam, mas também pelo seu funcionamento. Por outro lado, a

Visabeira Digital ficou com uma nova arquitectura. Esta arquitectura, devido ao grande dinamismo que apresenta, irá permitir à Visabeira Digital, não só desenvolver um novo tipo de funcionalidades (com funcionamento dual), mas também integrar dados entre os vários Sistemas de Informação de uma forma mais simples e centralizada (através do Sistema Integrador/Servidor).

Para além dos objectivos propostos, a funcionalidade de "Assistência remota" demonstrou o seu valor, sendo uma ferramenta bastante utilizada pela equipa de *help-desk*. Uma vez que esta permite um rápido acesso à máquina cliente que solicitou o pedido, os utilizadores finais são atendidos e esclarecidos mais rapidamente.

Para rematar, há que referir que uma vez que o tempo destinado para a implementação deste projecto foi escasso, não houve a oportunidade de efectuar testes de usabilidade e de aceitação. Contudo, mediante contactos directos e pontuais com os utilizadores, apercebemo-nos do contentamento e satisfação geral demonstrada.

Conclui-se ainda, que a solução implementada enquadra-se bem no assunto desta dissertação "Arquitectura Dual de Integrador de SIs Heterogéneos com Sincronização".

5.2 Balanço Geral

Para fazer um balanço geral de todo o cenário que envolve a arquitectura proposta, foi elaborada uma análise de *Strengths, Weaknesses, Opportunities and Threats* (SWOT). A análise SWOT é frequentemente realizada em torno de factores críticos de sucesso e tem como objectivo principal avaliar as vantagens concorrenciais e os pontos fracos da solução para a organização. Esta análise permitirá não só posicionar melhor o sistema proposto na sua área de negócio, mas também determinar novos nichos de mercado ou novos produtos, de forma a maximizar as forças, aproveitar as oportunidades para combater as ameaças, minorar as fraquezas e produzir um valor acrescentado para a empresa [WEIHRICH, 1982].

Desta forma, a análise SWOT elaborada proporciona um resumo estruturado das características que favorecem ou não esta solução a nível interno e externo. Recorrendo à análise apresentada na Ilustração 68, verificamos que existe um grande conjunto de pontos relativos a forças e oportunidades que revelam a importância do sistema proposto para a melhoria dos processos internos e externo da empresa.

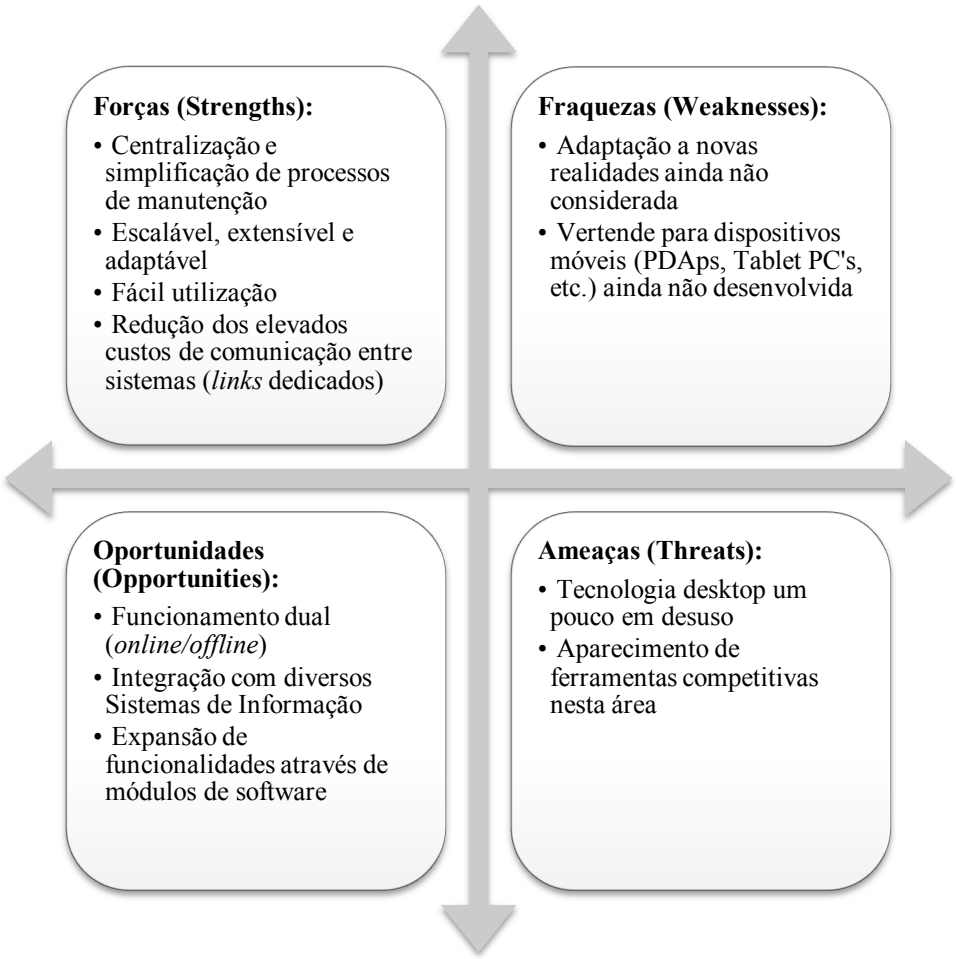

Ilustração 68 – Análise SWOT relativa à solução proposta nesta dissertação.

Iniciando a discussão pelas oportunidades, e relativamente a cada uma, há a considerar o seguinte:

- O funcionamento dual (*online*/*offline*) da arquitectura proposta assegura a disponibilidade do Sistema Cliente, permitindo que os utilizadores não tenham quebras de produtividade, anteriormente causadas, no caso da Viatel, pelas falhas de ligação entre os armazéns e o sistema central. Esta característica permitirá actuar sobre a primeira ameaça enumerada, uma vez que para garantir um funcionamento dual é necessário o recurso à tecnologia *desktop*, quer para armazenar informação localmente, quer para processar essa mesma informação (na ausência de uma ligação ao sistema central).
- A integração com os diversos Sistemas de Informação permite que a arquitectura tenha um carácter centralizador e integrador, uma vez que todos os processos rotineiros de sincronização e integração de dados, entre os diversos Sistemas de Informação, estão centralizados num único ponto, o Sistema Integrador/Servidor.
- A expansão de funcionalidades através de módulos de *software* que permitem a expansão do sistema proposto a novos negócios e empresas. Esta é a oportunidade identificada como sendo a que directamente irá combater a segunda ameaça identificada (o aparecimento de ferramentas competitivas). Isto porque, cada vez mais, surgem novas ferramentas competitivas nesta área. Contudo, poucas são as ferramentas que permi-

tem a sua expansão a novas áreas de negócio (completamente distintas) através do desenvolvimento e incorporação de novos módulos de *software*, não esquecendo a respectiva integração com o Sistema de Informação.

De igual modo, e uma vez que evitar as ameaças nem sempre é possível, deve ser feito um planeamento com o objectivo de prever, enfrentar e minimizar os efeitos das ameaças identificadas. De um modo geral, as oportunidades atrás referidas vão intensificar todas as forças apresentadas na Ilustração 68, já que:

- Centralizam e simplificam os processos de manutenção, uma vez que a arquitectura está centralizada em torno do Sistema Integrador/Servidor. Também a manutenção dos Sistemas Clientes é feita de forma automática, através da sincronização de dados, ou da actualização da aplicação informática, instalada por *ClickOnce*.
- Reforçam o carácter dinâmico da arquitectura, uma vez que esta é facilmente escalável (através do aumento do número de servidores), extensível (por módulos de *software*) e adaptável (quer a novas áreas de negócio, quer a novos requisitos).
- Permitem a redução dos custos elevados de comunicação (*links* dedicados) suportados pela Viatel, entre os seus armazéns e o sistema central. Esta redução é possível graças ao funcionamento dual proporcionado pela arquitectura proposta.

5.3 Orientações para trabalhos futuros

A implementação da arquitectura proposta neste documento teve uma grande aceitação pela Viatel, que demonstrou junto das outras empresas o seu agrado e satisfação para com a nova solução. Com este testemunho, outras empresas, de áreas de negócio diferentes, demonstraram o seu interesse nesta solução. De imediato, abordaram a possibilidade de desenvolver uma funcionalidade que permitisse cruzar a informação sobre o registo de picagens de ponto dos colaboradores e, com base nesse cruzamento e no mapa obrigatório de férias, proceder ao lançamento e aprovação massiva de férias. Esta funcionalidade seria utilizada para o lançamento automático de dias de férias comuns a todos os colaboradores, desde que estes não tenham registado picagem de ponto. O lançamento destes registos seria efectuado no respectivo módulo de Recursos Humanos do SAP. Com a arquitectura proposta, a implementação desta funcionalidade seria extremamente simples. Para esta implementação, apenas seria necessária a criação de um módulo de *software* que, no modo *online*, recolhesse dos respectivos Sistemas de Informação os registos de picagens de ponto dos colaboradores e a informação do mapa obrigatório de férias. Efectuada a recolha e o cruzamento da informação, esta seria apresentada ao utilizador para que este a possa validar. Após validação da informação, a funcionalidade estabeleceria a integração dos dados directamente no módulo de Recursos Humanos do SAP (o processo de integração de dados está descrito na secção 4.4). Esta funcionalidade ficou agendada como futuro desenvolvimento de *software* a integrar no sistema proposto neste projecto.

Como trabalho futuro e de continuação desta dissertação, existem algumas direcções que podem ser seguidas de forma a valorizar a solução desenvolvida.

Uma vez que alguns dos projectos constituintes dos Sistemas BeOn dão suporte à mobilidade através de soluções para PDA, seria importante estender o Sistema Cliente existente na arquitectura proposta também a dispositivos móveis, tais como: PDA's, Tablet PC's, etc.

Numa vertente tecnológica, a serialização de formulários poderia trazer algumas vantagens em funcionalidades mais básicas. Esta serialização iria permitir a criação dinâmica de formulários e posteriormente proceder ao armazenamento do próprio formulário e dos dados que este contém. Digamos que o seu funcionamento seria semelhante a um formulário criado no Microsoft InfoPath. A vantagem em relação à actual implementação deve-se ao facto de as novas funcionalidades (formulários) poderem ser definidas e representadas através de "modelos" no formato XML, com as respectivas validações dos dados. Assim, ao abrir uma funcionalidade em vez de carregar o binário de uma DLL, seria carregado o "modelo" através da sua definição XML. O utilizador preencheria esse modelo que posteriormente era gravado como um novo registo na Base de Dados. Contudo, a aplicação desta solução poderá levantar alguns problemas face à complexidade de determinadas funcionalidades. Outro obstáculo prende-se com o facto de a evolução para esta vertente tecnológica passar pela reimplementação da aplicação cliente na tecnologia WPF da Microsoft, abordada na subsecção 4.1.1.

Relativamente ao modelo lógico de dados proposto, e prevendo a futura integração de funcionalidades de outras áreas de negócio e o consequente aumento do número de tabelas de apoio necessárias ao funcionamento no modo *offline*, seria importante interligar as tabelas de apoio (entidade "AuxiliarData") com as funcionalidades que necessitam dessa informação (entidade "ConfigFunctionalities"). Analisando o modelo lógico de dados proposto na Ilustração 26, verificamos que a entidade "AuxiliarData" não está relacionada com nenhuma outra tabela. Para o sistema implementado, as tabelas de apoio são identificadas por um nome e pela sua estrutura da organizacional (Firma, Organização de Compras, Centro Logístico e Depósito). Desta forma, a estrutura organizacional será a ligação entre as tabelas de apoio disponíveis e os acessos de um determinado utilizador (entidade "ConfigUsersOrganizationAccess"), sendo sincronizadas para o Sistema Cliente apenas as tabelas de apoio que estabelecem esta ligação. Rapidamente nos apercebemos que com o aumento do número de tabelas de apoio, poderão ser sincronizadas para o Sistema Cliente algumas tabelas de apoio desnecessárias (uma vez que, como foi definido pela Visabeira Digital, o utilizador possui o respectivo acesso à estrutura organizacional), contudo estas poderão nunca vir a ser utilizadas (caso o utilizador não tenha acesso às funcionalidades que recorrem a estas tabelas de apoios). Assim sendo, a implementação desta ligação será uma vantagem, na medida em que permitirá reduzir o tráfego entre o Sistema Integrador/Servidor e o Sistema Cliente, e ainda o número de tabelas de apoio existentes em cada Sistema Cliente.

Uma outra melhoria identificada com o possível crescimento do número de funcionalidades prende-se com o requisito deste projecto de bloquear o acesso à aplicação quando esta está a operar no modo *offline*, característica abordada na subsecção 4.3.1. Com o aumento do número de funcionalidades disponíveis faz todo o sentido que o bloqueio seja imposto à funcionalidade e não à aplicação. Assim o controlo de acesso será mais flexível e permitirá proteger realmente apenas as funcionalidades que interessam. Este controlo de acesso pode ser implementado de duas formas distintas. A primeira, alterando a aplicação base para que esta bloqueie os botões que permitem o

acesso às funcionalidades cujo período temporal já tenha expirado. Nesta situação, teria ainda que ser armazenada localmente a informação do período temporal referente a cada funcionalidade. Na segunda situação apenas seria necessário implementar a validação do período temporal em cada uma das funcionalidades que se pretenda proteger. Esta alteração não carecia de qualquer alteração da Base de Dados do Sistema Cliente ou da própria aplicação base, uma vez que seria a própria funcionalidade a consultar a informação disponível ("último acesso ao sistema" referido na subsecção 4.3.1), e validar o seu estado, impedindo ou não a sua utilização. Após implementação do controlo de acesso, numa determinada funcionalidade, e da sua sincronização para o Sistema Cliente, esta ficaria instantaneamente protegida. A simplicidade deste desenvolvimento reforça as potencialidades que a arquitectura proposta apresenta.

Por último, e não menos importante, será a alteração da funcionalidade "Assistência remota" para permitir o acesso às máquinas que não estão na rede interna do Grupo Visabeira, uma vez que a funcionalidade actual não o permite. A solução deste problema poderá passar por estabelecer uma ligação VPN instantânea entre a máquina do utilizador e a máquina da equipa de *help-desk*. Esta melhoria irá permitir que a equipa de *help-desk* consiga estabelecer uma ligação remota a qualquer máquina e sem a necessidade de utilizar outro *software*.

Bibliografia

ALMEIDA, Ricardo Gabriel Soares Fernandes de – "Integração de Informação por Migração em Sistemas Distribuídos e Heterogéneos". Vila Real: Universidade de Trás-os-Montes e Alto Douro. 2009. Dissertação de Mestrado.

COSTA, João Gonçalves da – "Offline Web Applications: Enabling offline execution on the WOW! Product". Porto: Faculdade de Engenharia da Universidade do Porto. 2008. Dissertação de Mestrado.

HART, Johnson M.; ROSENBERG, Barry – "Client/Server Computing for Technical Professionals: Concepts and Solutions". Addison-Wesley. 1995. ISBN 0-201-63388-4

JANKOWSKI, Alex; GEORGI, Henrik – "Ofline Framework for Smart Clients". Technical University of Denmark. 2006. Dissertação de Mestrado.

MARTINS, Victor Manuel Moreira – "Integração de Sistemas de Informação: Perspectivas, normas e abordagens". Guimarães: Universidade do Minho. 2005. Dissertação de Mestrado.

SEIDEL, Jan – "Optimization Strategies for Client-Server Mapping and Data Distribution Schemes in a Parallel Memory File System". Fachhochschule Bonn-Rhein-Sieg – University of Applied Sciences. 2007. Dissertação de Mestrado.

Referências

[ACCENTURE, 2011] Página Web da empresa Accenture
URL: http://www. Accenture.com, acedido em: Janeiro de 2011

[AdobeFlex, 2011] Adobe Flex
URL: http://www.adobe.com/products/flex/, acedido em: Maio de 2011

[AdobeforForce, 2010] Adobe Flash Builder for Force.com
URL: http://developer.force.com/flashbuilder, acedido em: Novembro de 2010

[AdobeRIA, 2011] Adobe RIA
URL: http://www.adobe.com/resources/business/rich_internet_apps/, acedido em: Abril de 2011

[ALBAHARI, 2011] ALBAHARI, Joseph – "Threading in C#". O'Reilly Media, Inc. 2011. Livro electrónico.

[ANDERSON, 2008] ANDERSON, J. Ross – "Security Engineering". 2ª Edição. Indiana: Wiley. 2008. ISBN 978-0-470-06852-6. Cap. 5.

[Appforce, 2010] Application Development with the Force.com Cloud Computing Plataform - Appforce
URL: http://www.salesforce.com/eu/platform/appforce.jsp, acedido em: Novembro de 2010

[BALLINGER, 2003] BALLINGER, Keith – ".NET Web Services: Architecture and Implementation". Addison-Wesley. 2003. ISBN 0-321-11359-4

[BARFIELD, 1993] BARFIELD, Lon – "The User Interface: Concepts & Design". Addison-Wesley. 1993. ISBN 0-201-54441-5

[CALLAHAN, 2008] CALLAHAN, C. A. – "Mastering: Windows SharePoint Services 3.0". Indiana: Wiley. 2008. ISBN 978-0-470-12728-5

[ClickOnce, 2011] ClickOnce Security and Deployment
URL: http://msdn.microsoft.com/en-us/library/t71a733d(VS.100).aspx, acedido em: Junho de 2011

[CLR, 2011] Using CLR Integration in SQL Server 2005
URL: http://msdn.microsoft.com/en-us/library/ms345136(v=sql.90).aspx, acedido em: Maio de 2011

[CLRExternalWebService, 2011] CLR Stored Procedure Calling External Web Service
URL: http://davidhayden.com/blog/dave/archive/2006/04/25/2924.aspx, acedido em: Maio de 2011

[COULOURIS et. al., 2001] COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim – "Distributed Systems: Concepts and Design". 3ª Edição. Inglaterra: Addison-Wesley. 2001. ISBN 0201-61918-0

[DAMAS, 2005] DAMAS, Luís – "SQL Structured Query Language". 6ª Edição. Lisboa: FCA. 2005. ISBN 972-722-443-1

[Database.com, 2010] Application Development with the Force.com Cloud Computing Platform - Database.com
URL: http://www.salesforce.com/eu/platform/database.jsp, acedido em: Novembro de 2010

[DeveloperForce, 2010] Developer Force: Salesforce.com & Force.com Developer Resources
URL: http://developer.force.com, acedido em : Novembro de 2010

[DevExpress, 2011] Página Web da empresa Developer Express
URL: http://www.devexpress.com/, acedido em: Junho de 2011

[EmbeddingSQLServer, 2011] Embedding SQL Server Express into Custom Applications
URL: http://msdn.microsoft.com/en-us/library/bb264562(v=sql.90).aspx, acedido em: Maio de 2011

[Force.com, 2010] Application Development with the Force.com Cloud Computing Platform
URL: http://www.salesforce.com/eu/platform,
Acedido em: Novembro de 2010

[GRAVE, 2002] GRAVE, Ana Cristina da Conceição – "Sistemas ERP em Ambiente Móvel". Faculdade de Ciências da Universidade de Lisboa. 2002. Dissertação de Mestrado.

[GrupoVisabeira, 2010] Página Web do Grupo Visabeira, SGPS, S.A.
URL: http://www.grupovisabeira.com, acedido em: Novembro de 2010

[HostingWCFServices, 2011] Hosting and Consuming WCF Services
URL: http://msdn.microsoft.com/en-us/library/bb332338.aspx, acedido em: Maio de 2011

[HUGHES et. al, 1997] HUGHES, Conrad; WINSLOW, Maria; HUGHES, Merlin e SHOFFNER, Michael – "Java Network Programming". Pearson Education. 1997.

[Java, 2011] Java
URL: http://www.java.com/en/about/, acedido em: Maio de 2011

[JavaFX, 2011] Java FX
URL: http://javafx.com/, acedido em: Maio de 2011

[KLEIN, 2006] KLEIN, Scoot – "Professional SQL Server 2005 XML". Indianapoles: Wiley Publishing, Inc. 2006. ISBN 0-7645-9792-2

[KRETSCHMER e WEISS, 1996]	KRETSCHMER, Rüdiger e WEISS, Wolfgang – "Developing SAP's R/3 applications with ABAP/4". Sybex. 1996.
[LinkingServers, 2011]	Linking Servers URL: http://msdn.microsoft.com/en-us/library/ms188279.aspx, acedido em: Maio de 2011
[LOPES e RAMALHO, 2005]	LOPES, Carlos Jorge; RAMALHO, José Carlos – "Web Services - Aplicações Distribuídas sobre Protocolos Internet". FCA. 2005. ISBN: 978-972-722-421-0
[MAGALHÃES, 2008]	MAGALHÃES, Alberto – "SQL Server 2008: Curso Completo". FCA. 2ª Edição. 2008. ISBN: 978-972-722-684-9
[Managed e Unmanaged Code, 2003]	An Overview of Managed/Unmanaged Code Interoperability URL: http://msdn.microsoft.com/en-us/library/ms973872.aspx Acedido em: Maio de 2011
[MANES, 2003]	MANES, Anne T. – "Web Services: A Manager's Guide. Addison-Wesley Pearson Education, Inc. 2003. ISBN 0-321-18577-43
[MSSharePoint2010, 2010]	Página Web do Microsoft SharePoint 2010 URL: http://sharepoint.microsoft.com, acedido em: Dezembro de 2010
[MSSharePointWorkspace, 2010]	Página Web do Microsoft SharePoint Workspace URL: http://office.microsoft.com/pt-pt/sharepoint-workspace-help/, acedido em: Dezembro de 2010
[MSSQLServerCompact, 2011]	Microsoft SQL Server 2008 Compact URL: http://www.microsoft.com/sqlserver/2008/en/us/compact.aspx, acedido em: Abril de 2011
[MSSQLServerEnterprise, 2011]	Microsoft SQL Server 2008 R2 Enterprise URL: http://www.microsoft.com/sqlserver/en/us/editions/enterprise.aspx, acedido em: Abril de 2011
[MSSQLServerExpress, 2011]	Microsoft SQL Server 2008 R2 Express URL: http://www.microsoft.com/sqlserver/2008/en/us/express.aspx, acedido em: Abril de 2011
[MSSyncFramework, 2011]	Página Web da Microsoft Sync Framework URL: http://msdn.microsoft.com/pt-br/library/bb902827.aspx, acedido em: Maio de 2011
[MUSCIANO e KENNEDY, 2000]	MUSCIANO, Chuck e KENNEDY, Bill – "HTML & XHTML:The Definitve Guide". 4ª Edição. O'Reilly & Associates, Inc. 2000. ISBN 0-596-00026-X.
[NIELSEN, 1993]	NIELSEN, Jakob – "Usability Engineering". São Francisco: Morgan Kaufmann. 1993. ISBN 0-12-518406-9
[NOEL e SPENCE, 2007]	NOEL, Michael e SPENCE, Colin – "Microsoft SharePoint 2007: Unleashed". Indiana: Sams. 2007. ISBN 0-672-32947-6

[OLE Automation, 2011] OLE Automation Stored Procedures (Transact-SQL)
URL: http://msdn.microsoft.com/en-us/library/ms190501.aspx, acedido em: Maio de 2011

[Oracle10gXE, 2011] Oracle Database 10g Express Edition
URL: http://www.oracle.com/technetwork/database/express-edition/overview/index.html, acedido em: Abril de 2011

[Oracle11gR2, 2011] Oracle Database 11*g* Release 2
URL: http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html, acedido em: Abril de 2011

[SalesForce, 2010] Página Web da empresa SalesForce
URL: http://www.salesforce.com, acedido em: Novembro de 2010

[SAP, 2011] Página Web da empresa SAP
URL: http://www.sap.com, acedido em: Janeiro de 2011

[SAPBC, 2011] SAP Business Connector Fact Sheet.
URL: http://www.sap.com/solutions/technology/pdf/50033842.pdf, acedido em: Junho de 2011

[SAPMM, 2010] Projecto de implementação SAP R/3:Materials Management. 2010 Disponibilizado pelo Grupo Visabeira.

[SharePointFarm, 2011] SharePoint Farm configuring and deployment
URL: http://sharepointmagazine.net/articles/best-practices-of-sharepoint-farm-configuring-and-deployment-part-1-architectural-and-logical-planning, acedido em: Fevereiro de 2011

[Silverlight, 2011] Silverlight.NET
URL: http://www.silverlight.net, acedido em: Maio de 2011

[Telerik, 2011] Telerik
URL: http://www.telerik.com/, acedido em: Junho de 2011

[VisualStudio, 2011] Overview of Visual Studio 2010 Professional
URL: http://www.microsoft.com/visualstudio/en-us/products/2010-editions/professional/overview, acedido em: Junho de 2011

[VMforce, 2010] Application Development with the Force.com Cloud Computing Plataform - VMforce
URL: http://www.salesforce.com/eu/platform/vmforce.jsp, acedido em: Novembro de 2010

[WCFServices, 2011] Windows Communication Foundation Services
URL: http://msdn.microsoft.com/en-us/library/aa480190.aspx#introt_topic1, acedido em: Maio de 2011

[WebServices, 2011] Web Services
URL: http://msdn.microsoft.com/en-us/library/ms950421.aspx, acedido em: Maio de 2011

[WEIHRICH, 1982] WEIHRICH H. – "The TOWS Matrix: A Tool for Situational Analy-

sis". Journal of Long. 1982.

[WILL et. al., 1998] WILL, Liane; HIENGER, Christiane; STRABENBURG, Frank e HIMMER, Rocco – "SAP R/3 Administration". Addison-Wesley. 1998. 0-201-92469-2.

[WindowsForms, 2011] Windows Forms Applications
URL: http://msdn.microsoft.com/en-us/library/ms644558(v=VS.85).aspx, acedido em: Maio de 2011

[WPF, 2011] Windows Presentation Foundation
URL: http://msdn.microsoft.com/en-us/library/ms754130.aspx, acedido em: Maio de 2011

[XAML, 2011] XAML Overview (WPF)
URL: http://msdn.microsoft.com/en-us/library/ms752059.aspx, acedido em: Maio de 2011

[XMLBestPratices, 2011] XML Best Practices
URL: http://msdn.microsoft.com/en-US/library/ms187508(v=SQL.90).aspx, acedido em: Março de 2011