Universidade de Évora
Honesto Estudo com Longa Experiência Misturado
Departamento de Informática

Mestrado em Engenharia Informática

Business Intelligence em sistemas de apoio à gestão de frotas

Análise de tecnologias e metodologias

Paulo Daniel Caneira Andrade Pires

Orientador: Salvador Pinto Abreu

Setúbal, 25 de Novembro de 2010

Titulo: Business Intelligence em sistemas de apoio à gestão de frotas

Análise de tecnologias e metodologias

Autor: Paulo Daniel Caneira Andrade Pires

Orientador: Professor Doutor Salvador Pinto Abreu

Setúbal, 25 de Novembro de 2010

AGRADECIMENTOS

Quero deixar o meu agradecimento em primeiro lugar à empresa COM-UT Technologies Lda., da qual sou colaborador, por me ter proporcionado os recursos e infra-estruturas para desenvolver o projecto que serviu de caso estudo para esta tese, nomeadamente aos seus proprietários Richard Lafiton e Michaël Memeteau. Quero agradecer também aos meus colegas de trabalho e amigos Paulo Fournier, Joaquim Serafim, pelo apoio demonstrado no desenvolvimento deste projecto. Ao colega Pedro Cruz Rato pela ajuda com os seus conhecimentos de comercial, relativamente à parte de negócio comercial.

À Universidade de Évora pela formação tanto a nível profissional como pessoal e em especial ao Professor Doutor Salvador Pinto Abreu, meu orientador por parte desta instituição, pela sua disponibilidade, apoio e compreensão na escolha por mim efectuada para o tema desta dissertação e na realização desta tese.

Agradeço de forma sentida a todos os meus amigos que sempre me apoiaram, ajudaram e me deram motivação extra para seguir com este projecto e superar as dificuldades encontradas. Sem detrimento de ninguém pois todos tiveram a sua importância e sabem quem são, quero agradecer em particular a Paulo Fidalgo, Filipe Vieira, Nuno Miranda e Zé Duarte pelo apoio, motivação e entreajuda na realização deste projecto.

Para finalizar o meu principal e mais sentido agradecimento vai para a minha família, o meu pai Daniel Joaquim A. Pires e a minha mãe Maria de Lourdes Henriques C. A. Pires pois eles são a razão daquilo que sou hoje, a eles agradeço a formação moral e os valores pelos quais me rejo, o esforço que fizeram para que hoje tenha uma licenciatura fundamental para a construção do meu futuro, por acreditarem em mim e por estarem sempre presentes. Também às minhas irmãs Miriam Pires e Sara Andrade pela força, confiança e apoio que sempre me deram e às minhas avós pela força e motivação.

A todos estes e todos os meus amigos um MUITO OBRIGADO por tudo.

Resumo

O objecto de estudo desta tese de mestrado surgiu da necessidade de dar resposta a uma proposta para uma solução de business intelligence a pedido de um cliente da empresa onde até à data me encontro a desempenhar funções de analista programador júnior. O projecto consistiu na realização de um sistema de monitorização de eventos e análise de operações, portanto um sistema integrado de gestão de frotas com módulo de business intelligence. Durante o decurso deste projecto foi necessário analisar metodologias de desenvolvimento, aprender novas linguagens, ferramentas, como C#, JasperReport, visual studio, Microsoft SQL Server entre outros.

Abstract

Business Intelligence applied to fleet management systems – Technologies and Methodologies Analysis.

The object of study of this master's thesis was the necessity of responding to a proposal for a business intelligence solution at the request of a client company where so far I find the duties of junior programmer. The project consisted of a system event monitoring and analysis of operations, so an integrated fleet management with integrated business intelligence. During the course of this project was necessary to analyze development methodologies,

learn new languages, tools such as C #, JasperReports, visual studio, Microsoft Sql Server and others.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

API – Application Programming Interface (Interface de Programação de Aplicativos)

CLR – Common Language Runtime

GPS – Global Positioning System

GPRS – General Packet Radio Service

OLAP – Online Analytical Processing

PME – Pequena Média Empresa

POI – Point Of Interest (Ponto de Interesse)

SMS – Short Message Service (Serviço de Mensagem curta)

SQL – Structured Query Language (Linguagem de Consulta Estruturada)

Glossário

Aplicação stand alone – Aplicação que funciona de forma independente.

Business Intelligence – Processo de recolha, organização, análise, partilha e monitorização de informações que oferecem suporte à gestão de negócios.

Canbus – Sistema bus standard desenhado para permitir a microcontroladores e dispositivos comunicarem entre si num veículo sem um computador anfitrião.

Data Warehouse – Uma forma de armazenar dados com um desenho que favorece a pesquisa para relatórios, análise de grandes volumes de informação e obtenção de informações chave para facilitar a tomada de decisão.

Query – Forma de interrogação a uma base de dados, através de uma sintaxe própria, no caso de query SQL.

Stored procedure – Conjunto de comandos SQL armazenado na base de dados e que pode ser chamado para efectuar as tarefas especificadas nesses comandos.

Telemetrias – Dados recolhidos através de sensores ou sondas e enviados para os sistemas de informação que vão depois interpretar estas informações.

Triggers – Código procedural que é executado automaticamente em resposta a um determinado evento numa tabela ou vista particular de uma base de dados.

1 – Introdução

1.1 – Apresentação do conteúdo da tese

Este trabalho teve como objectivo a análise de especificações, desenho, desenvolvimento e implementação de uma ferramenta de business intelligence solicitada por um cliente da empresa da qual sou colaborador. A solução solicitada consistia num sistema que monitorizasse o fluxo lógico dos eventos definidos para a operação diária dos veículos da empresa cliente e também um módulo de análise aos dados recebidos e processados por esta lógica. Seria um sistema de controlo do dia a dia de trabalho feito pelos veículos, através da monitorização de um conjunto de eventos pré-definidos para a gestão de frotas, com um módulo de análise (Business Intelligence) integrado. Teve também o objectivo de analisar e estudar as metodologias e ferramentas actualmente conhecidas e disponíveis tendo como base a aplicação desenvolvida para esta tese.

Este projecto integra a área de negócio da empresa que é a gestão de frotas, uma área em expansão no nosso mercado.

Hoje em dia existem muitas tecnologias disponíveis para atingir os principais objectivos da gestão de frotas e já há muito deixou de ser apenas uma ferramenta de localização dos veículos. São muitas as informações que se podem retirar do veículo que vão alimentar o sistema sobre o qual poderá ser depois efectuada uma análise que levará a optimizações de negócio da empresa

utilizadora destes sistemas. Não só a tecnologia a nível de hardware é vasta e bastante melhorada actualmente, como também as tecnologias de software e ferramentas de desenvolvimento evoluíram e nos são oferecidas muitas opções. Tendo em conta estes factos, esta tese foi desenvolvida não só para demonstrar uma aplicação prática de uma solução para responder a um pedido específico de um cliente, mas também para fazer uma análise das metodologias e ferramentas que existem.

1.2 – Guia do Leitor

Esta tese está organizada do seguinte modo:

Capitulo 2: apresenta o projecto, a realidade em que o mesmo se insere e para o qual foi desenvolvido e aspectos técnicos.

Capitulo 3: específica o projecto que foi desenvolvido como caso de estudo. Mostra alguns processos de desenvolvimento, metodologias utilizadas e o que consistiu o projecto em si.

Capitulo 4: evidencia as conclusões e resultados da análise feita às ferramentas e metodologias utilizadas e outras possíveis opções ao que foi utilizado.

Capitulo 5: contém as conclusões e denota o trabalho que ainda é necessário desenvolver.

2 – Enquadramento Geral

Este capítulo serve para enquadrar o leitor no negócio para o qual o projecto foi desenvolvido, bem como no que consiste este mesmo projecto.

2.1 – Introdução à gestão de frotas e panorama actual

2.1.1 – História e evolução da gestão de frotas

A gestão de frotas é a organização e gestão da frota de veículos motorizados de uma companhia, podem eles ser automóveis, carrinhas, camiões, helicópteros, aviões, barcos, etc.

A gestão de frotas (veículos) pode incluir uma vasta gama de funções tais como financiamento de veículos, manutenções, telemáticas de veículos (tracking e diagnostico), gestão de motoristas, gestão de combustível e gestão de saúde e segurança. [2]

A função da gestão de frotas permite que companhias em que a sua área de negócio depende do transporte, removam ou minimizem os riscos associados ao investimento em veículos, melhorar a eficiência, produtividade e redução dos seus custos globais de transportação e muito mais.

Nesta tese quando menciono gestão de frotas estou-me a referir a sistemas de gestão de frotas, que incorporam registo de dados, posicionamento por satélite e comunicação de dados para uma aplicação.

Como já referido em cima, a gestão de frotas engloba várias funções, que serão brevemente especificadas de seguida.

Localização de veículos:

Esta é a função mais básica de qualquer sistema de gestão de frotas. Esta componente é normalmente baseada em GPS, ou triangulação celular. A localização, direcção e velocidade do veículo são determinados pelo componente GPS e depois é transmitido por GPRS ou SMS, para a aplicação de Gestão de Frotas. Os métodos de transmissão de dados incluem ambos os meios, terrestre ou satélite. As comunicações por satélite são mais caras, mas imprescindíveis se queremos que a localização de veículos trabalhe em ambientes remotos sem interrupção. [2]

Diagnósticos Mecânicos:

Sistemas de gestão de frotas mais avançados conseguem-se ligar ao computador de bordo do veículo e recolher informação do mesmo, para disponibilizar ao utilizador. Detalhes como quilometragem, consumo de combustível, são recolhidos para um esquema estatístico global. [2]

Comportamento do condutor:

É possível através de dados variados, recolhidos do computador de bordo, criar um perfil do motorista que conduz esse veículo. [2]

Software de gestão de frotas

Estes softwares permitem aos utilizadores executarem uma série de tarefas específicas na gestão de alguns ou todos os aspectos relacionados à frota de veículos de uma empresa. Estes softwares dependendo das necessidades do cliente e da solução adquirida, permitem gerir as funções anteriormente especificadas e também fazer uma análise sobre os dados adquiridos, na forma de relatórios. [2]

Outras Funcionalidades:

Existem muitas mais funcionalidades que a gestão de frotas permite aos seus utilizadores, como por exemplo na área da segurança, é possível bloquear uma viatura à distância através de corte de corrente, desde que a mesma esteja equipada com dispositivos que o permitam.

Estes sistemas possibilitam que um utilizador remotamente evite que um motor tenha ignição, parar ou abrandar um veículo. Depois de parado o veículo alguns sistemas bloqueiam os travões ou cortam a corrente do mesmo não o

deixando ligar novamente. Os sistemas de paragem remota podem ser integrados com um sistema de notificação remoto de pânico ou emergência. Numa emergência um motorista pode enviar um alerta através do pressionamento de um botão de pânico, que poderá estar localizado no painel do veículo ou num dispositivo móvel com o motorista, se este estiver nas proximidades do veículo.

Historicamente os sistemas de gestão de frotas tiveram início há décadas atrás. Os computadores de bordo emergiram nos anos 80, e cedo foram ligados a várias redes wireless por satélite ou terrestres. Hoje em dia com o avanço da tecnologia, a capacidade das redes móveis oferecem-nos a ligação online a custos razoáveis e a tecnologia de computação móvel dá-nos uma alta performance e também usabilidade. Todos estes componentes ligados dão-nos a capacidade de criar sistemas de gestão de frotas oferecendo às empresas que os adquiram a possibilidade de optimizar custos e gerir as frotas de veículos, o transporte de mercadorias, os motoristas, gastos inerentes às deslocações, como combustíveis, portagens, etc., entre outros. [1]

Segundo um estudo de mercado feito pela empresa independente de análise Berg Insight [1], o mercado da gestão de frotas está a entrar num período de crescimento que irá continuar pelos próximos anos.

Contudo este é um mercado que também é afectado pela crise económica global que se instalou nos últimos tempos, levando a que este crescimento seja

sentido de um modo mais lento, pois há uma baixa no investimento, o que faz com que a adopção de nova tecnologia se faça de um modo mais lento. [1]

O que se verifica no mercado é que as empresas estão a adiar projectos, e investimentos que já estavam perto da concretização, daí sentir-se um crescimento a um ritmo mais baixo neste período de crise. Um número cada vez maior de empresas mostra não ter capacidade financeira necessária para implementar de um modo eficaz uma solução de gestão de frotas. Contudo a busca, cada vez mais necessária, pela eficiência financeira na redução de custos irá ser a principal razão impulsionadora desta tecnologia. [1]

De acordo com esta mesma consultora [1] (Berg Insight), a taxa de crescimento da aquisição destes sistemas para a gestão de frotas das empresas vai continuar a crescer em 2010, mas a um passo mais lento do que inicialmente previsto.

Estas conclusões terão que ser adequadas a cada mercado, pois depende muito do desenvolvimento económico local. Em Portugal, está a verificar-se este abrandamento na adopção deste tipo de sistemas nas empresas e verificou-se o adiamento de projectos, durante o ano de 2009 e neste período inicial de 2010.

O número de sistemas de gestão de frotas em uso activo é previsto crescer a uma taxa anual de 20,5% de 1,1milhão de unidades no final de 2008 para 3,3 milhões por volta de 2013. A taxa de entrada na população total de veículos

comerciais propriedade de empresas é estimado crescer de 3,1% em 2008 para 9,3% em 2013. Estes valores são conclusões retiradas do estudo da consultora Berg Insight [1].

2.1.2 – Evolução da tecnologia utilizada para os sistemas de ajuda à gestão de frotas

Os sistemas de gestão de frotas evoluíram muito desde o seu início até aos dias de hoje, graças à evolução da tecnologia, do que nos oferece como resultado, da portabilidade, usabilidade e redução de custos. O que nos primórdios dos dispositivos de gestão de frotas seria impossível, devido ao alto custo dessa tecnologia ou mesmo às dimensões que o aparelho teria que ter, hoje em dia é uma realidade e cada vez mais novas opções serão possíveis. Por exemplo: inicialmente tínhamos aparelhos capazes de dar a localização e velocidade através do módulo GPS, o que já trouxe alguma mais-valia às empresas pois tinham a possibilidade de seguir os seus veículos, verificar rotas, saber paragens e viagens, mas olhando para o que existe hoje, eram sistemas muito básicos.

Hoje temos aparelhos capazes de nos dar, dados vindos do próprio computador de bordo do veículo, pois já é possível integrar os dispositivos com a centralina do veículo através de Canbus, que é um standard de bus para veículos desenhado para permitir a microcontroladores e dispositivos comunicar

entre si, num veículo, sem ser necessário um computador anfitrião. Ou seja através deste protocolo de comunicação, é possível ligar os dispositivos de GPS do sistema de gestão de frotas instalado no veículo, ao computador de bordo do mesmo e retirar a informação necessária que vai depois alimentar o sistema. Temos também telemetrias, através de sensores, como portas abertas/fechadas, ignição ligada/desligada, através da colocação de sondas de temperatura, saber qual a temperatura dentro dos reboques frigoríficos, consumos, nível de combustível no depósito, com a colocação de um botão de pânico receber um SOS do motorista, entre outras. É possível também efectuar acções no veículo remotamente, como cortar a ignição, de modo a que quando o veículo parar, o mesmo desliga e não volta a arrancar, por motivos de segurança.

É possível receber dados do motorista, através de um módulo com teclado e ecrã, onde se pode configurar, mensagens pré-definidas e formulários e receber a informação que se deseja da parte dos motoristas.

A definição de itinerários através de pontos de passagem previamente configurados, possibilita uma optimização dos percursos efectuados e aliado à segurança, é possível a nível aplicacional definir se o veículo está a seguir o percurso definido ou não, e caso não esteja, poderá indicar um roubo do veículo.

Estas informações, aliadas a uma base de dados estruturada para as necessidades específicas duma situação, podem dar outros detalhes também importantes e oferece a possibilidade de ter uma análise do negócio, que vai dar a capacidade de optimizar o mesmo e os recursos disponíveis.

Podem-se fazer optimizações a nível de consumos, através de um conceito utilizado hoje em dia o "ecodrive", com o seguimento do veículo pode-se fazer uma análise de que condutores andam em rotações mais elevadas, quais os percursos onde o consumo é mais elevado, entre outras.

2.1.3 - Vantagens que uma empresa pode tirar com um sistema destes?

São muitos e diversos os benefícios que uma empresa pode tirar com a utilização de um sistema de apoio à gestão de frotas. As vantagens vão desde a optimização de gastos, à segurança e o simples controlo.

Para uma empresa pequena, digamos uma empresa com uma frota de entre 5 a 10 viaturas, o investimento inicial é mais baixo devido ao número reduzido de veículos, no entanto poderá não ter a liquidez suficiente para investir num sistema mais complexo. Contudo é um investimento que trás os seus resultados e dá retorno a médio prazo, por isso é algo que se deveria ter em conta. As empresas médias e grandes têm uma capacidade de investimento superior e podem adoptar soluções mais complexas e com um nível de informação superior. Portanto, o bom destes sistemas é que existe muita diversidade de aparelhos no mercado, logo pode-se adaptar uma solução ao tipo de empresa. Com a solução mais básica já podemos ter várias informações, será um aparelho com a capacidade de programação de formulários, mensagens pré-

definidas e com entrada para 4 telemetrias, onde pode ser colocado ignição, portas, temperatura, etc.

Portanto no geral, uma empresa para além de poder seguir a sua frota por todo o mundo, pode ainda efectuar análises a nível de muitos factores do negócio. Como por exemplo descobrir em que clientes estão a perder mais tempo, se há gastos de combustível fora do normal, pode-se com isto detectar roubos de combustível ou definir novas rotas que possam reduzir estes consumos, tempos em viagem e parado, entre outros e cada vez há mais dados que se podem retirar destes sistemas. A nível de segurança é possível enviar um SOS para o gestor de frotas no caso de um assalto ao veículo, em que o mesmo pode imobilizá-lo através de um comando enviado ao veículo.

Portanto são muitas as vantagens que uma empresa pode retirar destes produtos, daí haver cada vez mais empresas, pequenas, médias ou grandes a investir neste tipo de sistema. A nível nacional temos por exemplo, o estado com a A.N.P.C, que gere a frota de helicópteros, algumas companhias de bombeiros e também algumas empresas de transporte de vários tipos, com frotas que vão desde os 10 veículos aos 500 veículos.

2.1.4 – Realidade das empresas alvo deste tipo de sistemas em Portugal

O tecido empresarial português das empresas alvo deste tipo de sistemas é constituído maioritariamente por micro empresas e PME's e são este tipo de empresas que constituem o grosso das vendas no nosso país.

O grau de necessidade das funcionalidades destes sistemas (desde a simples localização ao Ecodrive) é sempre proporcional ao tipo de negócio e número de veículos constituintes da frota. Tudo depende da capacidade de investimento da empresa, pois uma solução mais complexa é mais cara também.

De acordo com o comercial da nossa empresa, e baseando-se na sua experiência, muitas vezes independentemente do valor ou capacidade de investimento, passa muito também pela mentalidade do gestor da empresa, sendo que a mesma ainda tem que ser mais "trabalhada" ou mesmo esperar por novas gerações com outra abertura para as tecnologias.

Segundo a experiência de contacto com os clientes do departamento comercial da empresa, o investimento feito neste tipo de solução, regra geral, dando uma estimativa genérica pode-se dizer que em 85% dos casos é pago em 2anos ou menos. Isto é conseguido através do incremento da produtividade, optimização de custos inerentes ao negócio, como redução de km excessivos e em conjunto com este a optimização do consumo de combustível. Estes factos aliados a uma melhoria na metodologia do próprio trabalho, são uma realidade

que se verifica numa primeira análise de indicadores (positivos) após a aquisição desta tecnologia.

Daí a pertinência destes sistemas e tendo a possibilidade de obter soluções moldadas à realidade de cada empresa, pois apesar de haver vários denominantes comuns, cada negócio tem as suas características próprias, as mais valia serão muito melhores se a tecnologia se moldar ao negócio e não o contrário.

2.2 – Proposta para a aplicação

Este projecto foi realizado no âmbito de responder às necessidades específicas de negócio de um cliente.

Nesta secção irei descrever os pontos básicos solicitados pelo cliente para a aplicação, que será descrita mais à frente.

Olhando para o documento com as especificações era claro que esta aplicação seria um sistema de Business Intelligence, pois o objectivo final resumia-se a conseguir ter a capacidade de retirar relatórios, com cálculos e análises aos dados recolhidos e armazenados pela parte lógica da aplicação.

Portanto, existem duas partes principais, o módulo "mecânico", que recebe os dados enviados pelos dispositivos dos veículos e insere-os na base de

dados depois de serem tratados pelo motor lógico da aplicação. A segunda parte principal é o módulo de análise, ou de relatórios, que será o módulo da aplicação que irá extrair os dados e executar os cálculos pretendidos para a análise que se quer fazer com cada relatório. A esta aplicação demos o nome de "optLogic", daqui para a frente referida apenas como Lógica.

2.3 – Ferramentas Utilizadas

Para o desenvolvimento deste projecto as ferramentas e linguagens utilizadas foram as seguintes:

- SQL Server 2008 Enterprise Edition
- C# – Linguagem de desenvolvimento
- Visual Studio 2008 Professional Edition – IDE de desenvolvimento
- PHP+ExtJS (desenvolvimento do interface web-based)
- JasperReports
- iReports
- Php JavaBridge

3 – Caso de Estudo

Neste capítulo vou incidir sobre o projecto que serviu de base para a construção desta tese, os pressupostos para o mesmo, a análise, desenho e desenvolvimento.

3.1 – Especificações e pré-requisitos

As especificações e pré-requisitos dados pelo nosso cliente para esta aplicação foram os seguintes:

Especificações e pré-requisitos

A Lógica tem que seguir as seguintes regras:

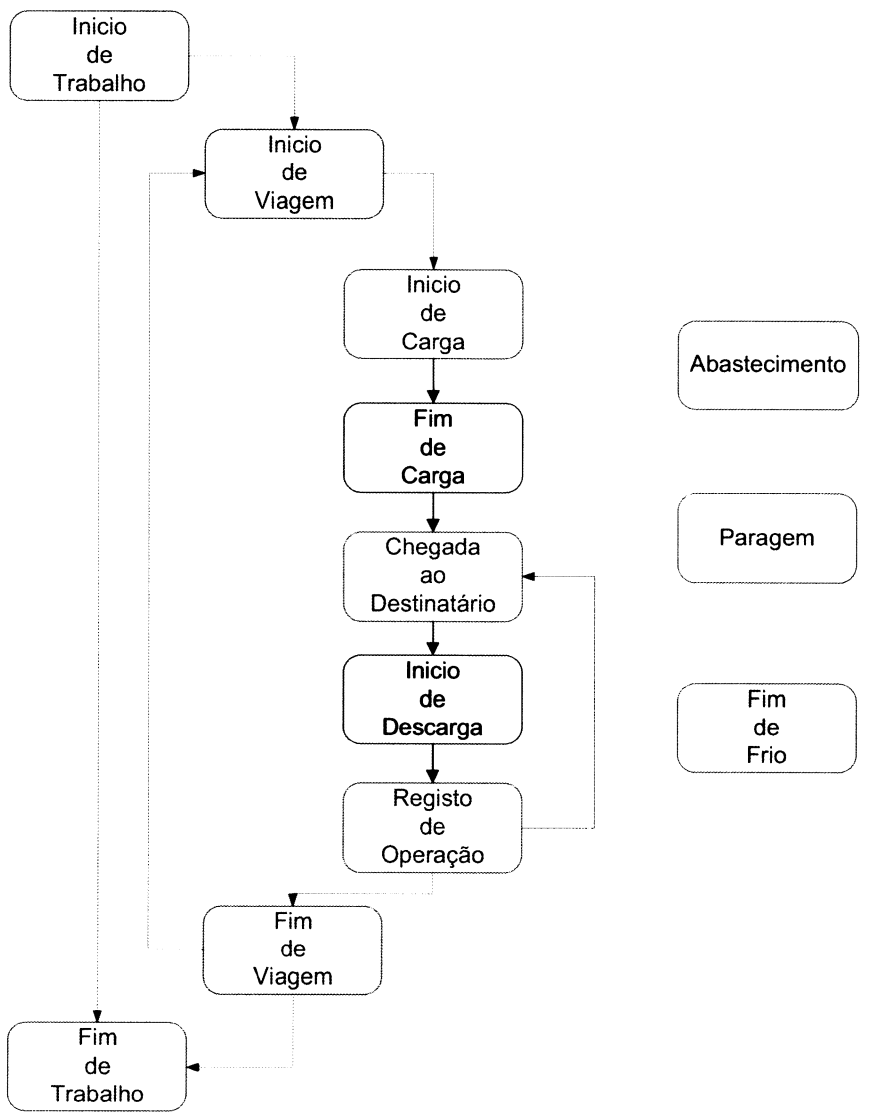
Existem inputs humanos (sequenciais e livres) e inputs automáticos (telemetrias).

Os inputs humanos sequenciais são feitos pelos motoristas que têm de seguir uma ordem pré-definida, que será descrita abaixo e os livres são inputs que podem ser efectuados em qualquer altura sem precisar de validação, ou seja podem surgir sem ter que haver um input precedente obrigatório.

Os inputs humanos podem ser formulários (em que o motorista insere valores em campos fixos) ou mensagens pré-definidas (mensagem já definida para a tecla de acção).

A sequência dos inputs pode ser observada no diagrama da figura 3.1 abaixo. Os inputs livres estão colocados a verde, os inputs que dependem de telemetrias obtidas através de sensores colocados no veículo, estão a azul e os restantes são os inputs humanos que têm que obedecer à sequência ilustrada.

Figura 3.1 – Diagrama da sequência dos inputs da lógica



Os formulários e mensagens pré-definidas para este sistema são descritas na tabela 3.1, abaixo.

Tabela 3.1 – Formulários e Mensagens pré-definidas

Nome	Tipo	Ordem de input
Inicio de trabalho	Formulário	1º
Inicio de viagem	Mensagem pré-definida	2º
Inicio de carga	Formulário	3º
Chegada ao destinatário	Mensagem pré-definida	4º
Registo de operação	Formulário	5º
Fim de viagem	Mensagem pré-definida	6º
Fim de trabalho	Mensagem pré-definida	7º
Abastecimento	Formulário	Independente
Paragem	Mensagem pré-definida	Independente
Fim de frio	Mensagem pré-definida	Independente

O objectivo do sistema é registar o dia de trabalho dos motoristas por isso os mesmos têm que enviar estes dados seguindo a mesma sequência que a

mostrada na tabela anterior. Contudo poderão existir ciclos internos que serão explicados de seguida, onde serão desenvolvidos cada um dos passos.

Inicio de Trabalho:

Indica que o motorista iniciou o seu trabalho diário. É um formulário que é enviado com os seguintes dados:

- Cliente: identificador do cliente da empresa ao qual o veiculo e motorista pertencem.
- Local de trabalho: identificador do armazém onde é iniciado o trabalho.
- Motorista: identificador do motorista que está no veículo.
- Km veículo: total de km's do veículo quando iniciado o trabalho.

Regras:

- Não pode existir "Inicio de trabalho" se o trabalho anterior não tiver sido finalizado.
- Só pode existir um Trabalho por dia.

Verificações/validações:

- Verificar se o código para o local de trabalho inserido existe e o número de motorista pertence a esse armazém.
- Verificar se os Km's inseridos são superiores aos km's constantes do último início de trabalho.

Caso haja alguma incoerência ou erro, deve ser enviado um alarme ao motorista.

Inicio de viagem:

Mensagem pré-definida que indica que o motorista está pronto a iniciar a viagem. Entende-se por viagem o bloco que contém a carga afecta a uma guia de transporte e as descargas nos clientes que constam dessa mesma guia. Uma viagem só contém uma guia de transporte e pode ter diversas descargas. A viagem é finalizada com o envio da mensagem pré-definida "Fim de Viagem". Um trabalho pode conter mais que uma viagem, ou mesmo nenhuma viagem, mas este é um caso especial.

Regras:

- Não pode haver início de viagem sem antes ter sido feito o início de trabalho para o dia corrente.
- Se não tiver sido fechada a viagem anterior, não pode haver um novo início de viagem.

Inicio de Carga:

O início de carga é um formulário enviado pelo motorista com os seguintes dados:

- Cliente: identificador do cliente da empresa ao qual o veiculo e motorista pertencem.
- Armazém: identificador do armazém onde a carga irá ser efectuada.
- Número de Guia: número identificativo da guia de transporte.
- Ajudante: campo de validação, onde se dirá se vai um ajudante ou não.

Pode existir mais do que um início de carga durante um período de trabalho, no entanto, só pode ser feito um início de carga por cada viagem.

Regras:

- Não pode ser feito início de carga sem antes ter sido feito um início de viagem.

Verificações:

- Verificar se a guia existe no armazém registado
- Verificar se a guia inserida não foi já registada noutra carga
- Verificar que o campo ajudante tem 1 – Sim ou 0 – Não

Chegada ao destinatário:

A chegada ao cliente é uma mensagem pré-definida que é enviada para registar a hora a que o veículo chegou a um destinatário. Podem existir tantas chegadas ao destinatário quantos destinatários estiverem registados na guia de transporte.

Regras:

- Para se fazer Chegada ao destinatário é necessário que tenha sido feito um inicio de carga anteriormente.
- Não pode ser feito chegada ao cliente sem ter sido fechada a entrega em aberto.

Verificações:

- A telemetria de abertura de portas, marca o início de descarga.
- O tempo entre a chegada ao cliente e a abertura de portas não deve exceder os 30minutos, caso isto aconteça é criado um alerta.
- Entre a chegada ao cliente e o registo de operação, que marca o final da descarga, não devem passar mais de 60min, caso isto ocorra é emitido um alerta.

Registo de operação:

O registo de operação é um formulário que indica a realização de uma entrega e é enviado com os seguintes campos:

- Número de destinatário: Indica o identificador do destinatário onde foi efectuada a entrega.
- Vasilhame: Indica se foi ou não recebido um vasilhame na entrega.
- Anomalia: Indica se houve alguma anomalia na entrega.

O número de registos de operações efectuadas é igual ao número de entregas que foram feitas.

Regras:

- O envio de registo de operação só será feito caso tenha sido feita uma chegada ao cliente, se não tiver ainda sido feito nenhum registo de operação no dia.
- Caso não seja o primeiro registo de operação para a viagem, pode ser feito mais que um registo de operação seguido, para salvaguardar as situações em que os locais de entrega são perto uns dos outros, de modo que só é feita uma chegada ao cliente para todos.

Verificações:

- Se o número de formulários enviados for superior ao número de entregas previstas na guia é emitido um alerta.
- Se o formulário for enviado com o campo Anomalia a "verdadeiro" é criado um alerta.

Fim de Viagem:

Isto é uma mensagem pré-definida que é enviada para registar o final de uma viagem ou seja todas as entregas foram efectuadas e o veículo vai parar ou iniciar uma nova viagem.

Regras:

- Não pode ser feito um fim de viagem sem ter sido feito um início de viagem inicialmente.
- Caso haja uma entrega em aberto, ou seja ainda não foi feito o registo de operação, não pode ser feito um fim de viagem.

Verificações:

- Se quando foi feito um fim de viagem o número de registos de operação for inferior ao número de entregas previstas na guia, é emitido um alerta.

Fim de Trabalho:

O fim de trabalho é uma mensagem pré-definida que é enviada quando o trabalho do foi terminado para o dia.

Regras:

- Não pode haver fim de trabalho sem ter sido feito um início de trabalho para o próprio dia.
- Se existir alguma viagem em aberto, não pode ser feito o fim de trabalho.

Fim de Frio:

O fim de frio é uma mensagem pré-definida que indica que o motorista vai desligar de forma premeditada o motor de frio do veículo.

Regras:

- Só é válido se existir uma viagem em aberto para o trabalho do dia corrente.

Verificações:

- Caso o motorista envie um fim de frio e não seja desligado o motor de frio num período de 2minutos, é enviado um alerta.
- Se o motor de frio for desligado sem que tenha sido enviado o aviso de Fim de Frio, é despoletado um alerta.

Abastecimento:

O abastecimento é um formulário que é enviado para identificar quando foi feito um abastecimento no veículo e tem os seguintes campos:

- Km's: Quilometragem marcada na altura do abastecimento.
- Litros: Quantidade de litros abastecidos no veículo.

Regras:

- Só é válido se existir um trabalho iniciado para o dia corrente.

Verificações:

- Se os quilómetros inseridos no formulário forem inferiores aos quilómetros enviados no último formulário de abastecimento do veículo emissor, é enviado um alerta.

Paragem:

Esta mensagem pré-definida indica que foi iniciada uma paragem longa, normalmente para sinalizar o período de almoço.

Regras:

- Só se existir um trabalho iniciado para o dia actual é que será validado esta mensagem.

Verificações:

- Se o tempo de paragem for superior a 2horas é emitido um alerta.

Para além dos inputs tempos também os alarmes que são os seguintes:

Temperatura:

- Se a temperatura das sondas no veículo sair fora do intervalo definido na guia de transporte, durante mais de 15minutos é despoletado um alarme.
- Se o veículo exceder a velocidade máxima definida de 93km/h é emitido um alarme.
- Se as entregas forem feitas fora das janelas horárias definidas para o destinatário em questão, será enviado um alarme.

Para qualquer um dos inputs efectuados, só poderão ser aceites se existir um trabalho aberto para o dia corrente. Caso já tenha sido fechado o dia de trabalho, este só volta a ser aberto, se a mensagem recebida for um Inicio de Viagem, em todos os outros casos o input é descartado.

Estas são as especificações para o sistema exigidas pelo cliente, para que depois possam ser feitas as análises necessárias para o negócio destes.

Estas especificações são as definições para o módulo da lógica, para o módulo de análise foram pedidos relatórios específicos, os quais não poderei detalhar neste documento, pois pertencem ao negócio específico do nosso cliente, razão pela qual não foi permitido o detalhe destes documentos.

Contudo e para que se tenha alguma noção do produto final dos dados que serão recebidos e tratados pela lógica, mencionarei de um modo mais geral estas análises.

O *Módulo de Análise,* como já mencionado anteriormente é composto por um conjunto de relatórios que retiram e tratam os dados da base de dados que foi criada para receber os mesmos através da lógica.

Até à escrita desta tese contam-se 13 relatórios específicos para este cliente, a adicionar aos relatórios base que são criados para todos os clientes da empresa COM-UT, contudo este número de relatórios pode crescer dependendo das necessidades dos clientes.

Estes relatórios são criados para analisar entregas, cargas, tempos de paragem, tempos de espera em armazéns e ou destinatários, para analisar o tempo despendido em cargas e descargas. É também feita uma análise a tempos de viagens, períodos de trabalho do motorista e gestão de abastecimentos. Estes relatórios contêm dados agregados como tempos totais e médias de tempos por veículos, por todos os veículos de um determinado armazém ou por todos os veículos do conjunto de armazéns pertencentes à empresa nossa cliente. Há também cálculos de consumos médios, médias e totais de distâncias percorridas.

Tendo em conta todos estes relatórios e as necessidades que os mesmos implicam, foi necessário ponderar bem no modelo de dados a adoptar para

responder da melhor forma a essas necessidades e também aos custos que o cliente estaria disposto a investir.

Estas escolhas e outras especificações mais técnicas deste projecto serão discutidas nas próximas secções.

3.2 – Arquitectura do sistema

Depois de analisar a fundo a proposta enviada pelo nosso cliente, foi altura de esboçar esquemas, escolher ferramentas, modelo de dados e a abordagem ao projecto em si. Nesta secção irei descrever o resultado desta análise e as escolhas tomadas e o porquê para a resolução do projecto que me foi colocado.

A nível de ferramentas de desenvolvimento e de servidor de base de dados, estive limitado nas minhas escolhas, uma vez que a empresa trabalha com um só servidor de base de dados, devido aos produtos de GPS que são adquiridos a uma empresa externa, com a qual a Com-UT tem uma parceria e às imposições do mercado nacional.

A empresa à qual o hardware e software de comunicação entre os dispositivos instalados nos veículos e a base de dados são comprados é a nossa parceira espanhola Datatronics. A Com-UT adquire os equipamentos e o software a esta empresa e depois desenvolve aplicações, funcionalidades em cima deste sistema base.

Daí as escolhas terem sido forçadas ao sistema base existente, que foi principalmente o servidor de base de dados, que é o Microsoft SQL Server 2008 no momento da escrita desta tese. A Datatronics desenvolve os seus sistemas para dois tipos de servidores de base de dados, que são eles SQL Server e Oracle, contudo devido à realidade do mercado português, o Oracle tinha custos de licenciamento demasiado elevados para o que os clientes estão dispostos a despender, em que o Oracle 11g Enterprise Edition custa um pouco mais do dobro que a licença de SQL Server 2008 Enterprise Edition e não trás tantas funcionalidades como o último.

Tendo então o servidor de base de dados, o passo seguinte foi definir que tipo de modelo de dados iria ser usado para um sistema destes. Analisando a fundo as especificações do cliente, principalmente o que iria recair no módulo de análise, que seria muita análise de negócio relacionado com o transporte, como análise de tempos, consumos, distâncias percorridas e isto em intervalos de tempo variáveis, como para um dia, um mês ou vários meses, a conclusão chegada foi que a solução mais adequada para responder a estas necessidades seria uma Data Warehouse. Contudo esta opção não foi a escolhida pois o cliente não estava disposto a investir o preço pedido para o desenvolvimento de um sistema integrado com uma Data Warehouse e também não queria esperar o tempo a mais que o desenvolvimento de um sistema destes requer para que seja robusto e fiável. Isto devido ao facto de que para avançar com um sistema

destes teria que ser investido um recurso da empresa para ter formação nesta área e para posteriormente fazer então o desenvolvimento.

Logo a opção escolhida teve que ser outra em que os custos para o cliente fossem mais baixos e o inicio do desenvolvimento fosse o mais rápido possível. A solução adoptada foi construir o sistema em cima de uma base de dados relacional, de modo a conseguir optimizar ao máximo a performance a nível de leitura de dados para os relatórios do módulo de análise.

Com isto definido o último passo foi decidir como seria implementado o sistema da Lógica e dos relatórios. Uma vez que a Lógica iria tratar os dados recebidos pelo sistema que foram enviados pelos dispositivos nos veículos, esta ferramenta foi então colocada à "escuta" destes inputs para depois tratá-los. Isto foi feito através de triggers colocados nas tabelas da base de dados principal, onde os dados em bruto vão cair e que quando accionados chamam a stored procedure que contém todo o código da Lógica e que irá tratar os dados recebidos e colocar na base de dados criada especificamente para este sistema.

O módulo de análise também consiste numa stored procedure feita em C#, que consoante o relatório escolhido vai correr o código específico para retirar os dados e fazer os cálculos necessários para a visualização do relatório em questão.

Modelo de dados:

Base de dados relacional normalizada de acordo com a interpretação e organização dos dados para o objectivo a atingir.

3.3 – Pormenorização do sistema

Como referido anteriormente o sistema desenvolvido, e olhando para o negócio da empresa no geral, consiste em módulos moldados às especificações do cliente em causa e que iram ser integrados na plataforma geral já existente e que é o ponto de encontro de todos os clientes. Ou seja existe uma aplicação geral com uma interface Web-based de seu nome SOMV (Sistema Online de Monitorização de Veículos), que é comum a todos os clientes e que contém para além de funcionalidades gerais a esses mesmos clientes, módulos específicos de cada um deles como unidade singular.

Portanto a aplicação criada e utilizada no âmbito desta tese consiste em dois módulos, "Lógica" e "Análise" como já referido anteriormente os quais foram integrados na plataforma SOMV e na qual se poderão ver os resultados directos do trabalho feito por estas partes. Temos então a parte "Lógica" que faz a monitorização em real-time dos dados que chegam e representa no frontend os eventos efectuados e os alertas gerados quando algo corre mal. E a parte de "Análise" que no frontend é então uma janela em que o utilizador pode seleccionar o relatório que pretende retirar, o(s) veículo(s) sobre o qual ou quais

pretende retirar o relatório, o período e o formato que quer exportar o mesmo.

As especificações deste projecto já foram descritas e as escolhas feitas de um modo geral também, portanto irei de seguida elaborar mais esses pontos de um modo progressivo, tal como abordado na realidade durante o desenvolvimento do projecto.

3.3.1 – Definição do modelo de dados

Depois de analisadas as especificações do cliente o primeiro passo foi definir o modelo de dados sobre o qual o sistema iria assentar.

Uma vez que o cliente não aceitou a proposta da implementação de um sistema de Business Intelligence com base numa data warehouse, a decisão foi construir uma base de dados relacional. Inicialmente pensou-se em desnormalizar esta base de dados, pois apesar de o sistema servir também para fazer uma monitorização de gestão de frotas, a principal funcionalidade que este cliente mais irá utilizar será a parte do módulo de análise, através dos relatórios criados para o efeito.

Algumas componentes que influenciam são o facto de esta ser uma frota com um número de veículos entre os 90 e 110. Cada veículo está programado para enviar dados de 1 em 1 minuto, sendo que para esta frota chegam em média cerca de 523 formulários e 588 mensagens pré-definidas por dia, isto sem

contar com as telemetrias, no total esta frota recebe em média cerca de 65.000 registos por dia de histórico. Portanto esta é a carga que o servidor tem que lidar, não esquecendo que existem outras frotas a enviar dados, pois este cliente tem o seu sistema instalado no servidor partilhado.

Posso afirmar com um valor aproximado que provavelmente 80% do acesso esta base de dados será de leitura, sendo que a escrita (inserção e actualização de dados) será feita para o módulo lógico apenas quando os dados chegam e para o módulo de análise praticamente não são feitos inserções. Logo para optimizar a leitura foi ponderada a hipótese de colocar este modelo de dados desnormalizado. Contudo tendo em conta os dados recolhidos, a quantidade de dados a ser pesquisada e o tamanho da base de dados, concluí que a performance em termos de rapidez de acesso a dados para leitura, não iria ter ganhos significativos, uma vez que os dados que iriam pesar mais nesta tarefa estão guardados na base de dados principal do sistema, a qual não poderia fazer alterações. A base de dados principal do sistema (dttDB) é a modulada pela empresa parceira Datatronics, para responder à estrutura do software que eles têm para receber os dados dos dispositivos e inserir nesta base de dados. Nesta base de dados o mais que podemos fazer é acrescentar tabelas que nos sejam úteis ou campos nas tabelas existentes. Daí ter escolhido criar uma base de dados só para este sistema que teria a liberdade para criar, modular ou refazer à medida das nossas necessidades.

O modelo de dados escolhido consiste no seguinte:

Os dados vão estar divididos em três grupos principais, dados relativos ao trabalho, relativos a viagens e a entregas, portanto temos uma tabela para cada um destes grupos. Na tabela 3.2, a baixo descrita podemos observar as tabelas principais.

Tabela 3.2 – Descrição das tabelas base da base de dados da aplicação

Tabela	Campos	Descrição
work	• id (Primary key) (identity) • id_fleet (int) • id_mobile (int) • date (varchar(10)) • [begin] (varchar(10)) • [end] (varchar(10)) • id_client (int) • id_cash (int) • id_driver (int) • mileage (double) • closed (bit)	• Identificador único do trabalho • Id. da frota • Id. do veículo • Data do trabalho • Hora de inicio • Hora do fim • Id. do cliente • Id. do armazém • Id. do condutor • Quilometragem inicial • Flag que indica se a viagem está fechada.
trip	• id (Primary Key) (identity) • id_work (Foreign Key) (int) • [begin] (datetime) • [end] (datetime) • load_begin (datetime) • load_end (datetime) • id_client (int) • id_cash (int) • n_guia (int) • helper (bit) • closed (bit) • iButton_1 (bit) • iButton_2 (bit)	• Identificador único da viagem • Id. do trabalho a que pertence a viagem • Data/hora do inicio viagem • Data/hora do fim viagem • Data/hora inicio de carga • Data/hora fim de carga • Id. do cliente • Id. do armazém da carga • Nº da guia de transporte • Flag(se tem ajudante) • Flag(se está fechada) • Id. do motorista1(iButton) • Id. do motorita2(iButton)
delivery	• id (Primary Key) (identity) • id_trip (Foreign Key) (int) • arrival (datetime) • departure (datetime) • unload_begin (datetime) • packaging (bit) • anomaly (bit) • closed (bit) • source (bit)	• Identificador único da entrega • Id. da viagem a que a entrega pertence • Data/Hora da chegada • Data/Hora de partida • Data/Hora de inicio de descarga • Flag(se tem vasilhame) • Flag(se tem anomalia) • Flag(se fechada) • Flag(identifica se é a primeira entrega de uma

		sequência seguida de entregas a diferentes clientes no mesmo local, ou seja apenas com uma chegada ao destinatário

Depois ligadas a estas temos as tabelas de status, que vão informar em cada tabela principal, quais os campos que foram introduzidos automaticamente, para posterior correcção dos mesmos, especificadas na tabela 3.3. Ou seja sempre que é detectado que não foi efectuado um passo, os dados em falta são inseridos automaticamente, com valores pré-definidos e é inserido na tabela de status correspondente qual o campo que foi adicionado desta forma, de modo a que se possa saber quais os campos que carecem de correcção por parte dos serviços administrativos da empresa.

Tabela 3.3 – Descrição das tabelas de status

Tabela	Campos	Descrição
work_status	• id (Primary Key) (identity) • id_work (Foreign Key) (int) • id_col (Foreign Key) (int) • type (smallint) • id_user (int) • timestamp (datetime) • ip (varchar(25))	• Identificador único • Identificador do trabalho a que pertence • Id. da coluna afectada • Tipo de anomalia • User que fez a alteração • Data/Hora da alteração • IP do user que fez a alteração
trip_status	• id (Primary Key) (identity) • id_trip (Foreign Key) (int) • id_col (Foreign Key) (int) • type (smallint) • id_user (int) • timestamp (datetime) • ip (varchar(25))	• Identificador único • Id. viagem pertencente • Id. da coluna afectada • Tipo de anomalia • User que fez a alteração • Data/Hora da alteração • IP do user que fez a alteração
delivery_status	• id (Primary Key) (identity) • id_delivery (Foreign Key) (int) • id_col (Foreign Key) (int) • type (smallint) • id_user (int) • timestamp (datetime) • ip (varchar(25))	• Identificador único • Identificador da entrega a que pertence • Id. da coluna afectada • Tipo da anomalia • User que fez a alteração • Data/Hora da alteração • IP do user que fez a alteração

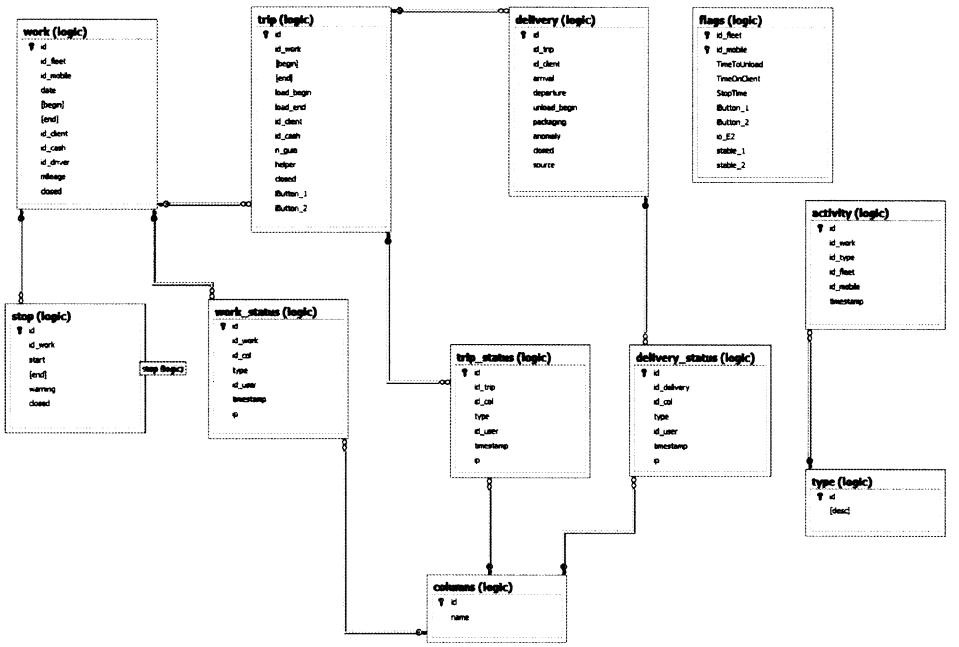
Por fim foram criadas as tabelas auxiliares, onde vão estar guardadas as actividades diárias, as flags para controlo de tempos de paragem, temporizadores etc., com a estrutura que se pode observar na tabela 3.4.

Tabela 3.4 – Descrição das tabelas auxiliares

Tabela	Campos	Descrição
flags	• id_fleet (Primary Key) (Foreign Key) (int) • id_mobile (Primary Key) (Foreign Key) (int) • TimeToUnload (datetime) • TimeOnClient (datetime) • StopTime (datetime) • iButton_1 (datetime) • iButton_2 (datetime) • io_E2 (datetime) • stable_1 (bit) • stable_2 (bit)	• Id. da frota a que pretence o veiculo • Id do veiculo afectado pela flag • Data/Hora inicio descarga • Data/Hora da chegada • Data/Hora da paragem • Data/Hora da sonda1 • Data/Hora da sonda2 • Paragem por telemetria • Flag(Sonda 1 estavel) • Flag(Sonda 2 estavel)
activity	• id (Primary Key) (identity) • id_work (int) • id_type (Foreign Key) (int) • id_fleet (int) • id_mobile (int) • timestamp (datetime)	• Id. único • Id. do trabalho pertencente • Id. do tipo de actividade • Id. da frota • Id. do veiculo • Data/Hora da actividade
type	• id (Primary Key) (identity) • [desc] (varchar(max))	• Identificador único • Descrição do tipo de actividade
stop	• id (Primay Key) (int) • id_work (Foreign Key) (int) • start (datetime) • end (datetime) • warning (bit) • closed (bit)	• Identificador único • Id. do trabalho pertencente • Data/Hora do inicio • Data/Hora do fim • Flag(paragem avisada) • Flag(Se fechada)

Este modelo de dados pode ser visualizado na figura 3.2 e também no Anexo A com maior legibilidade.

Figura 3.2 – Modelo de dados



Depois de construído o modelo de dados, foi altura de passar à implementação do módulo de lógica, que será descrito de seguida.

3.3.2 – Módulo da lógica

A primeira tarefa a executar foi delinear um plano para seguir durante a implementação deste módulo. Para tal construí um diagrama de Use Cases e diagramas de sequência e também para enviar ao cliente para que este confirmasse o plano de execução do sistema de lógica, que o mesmo especificou.

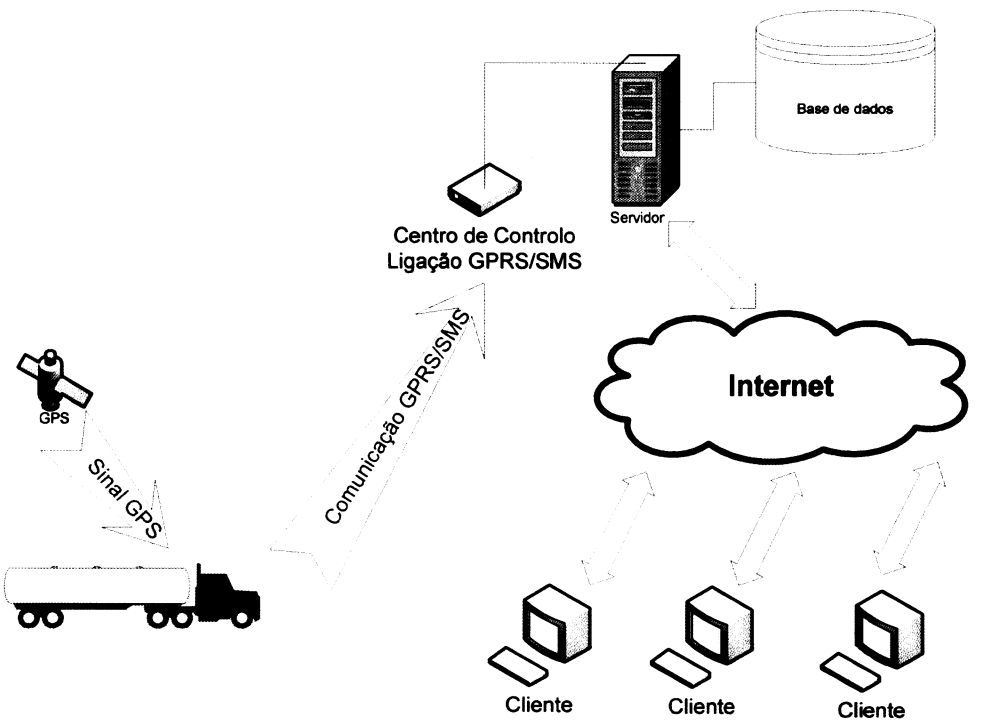
O diagrama de Use Case pode ser visualizado no anexo B e os diagramas de actividade nos anexos C a J.

Este módulo foi implementado numa Stored Procedure pois o objectivo foi fazer correr esta Stored Procedure quando os inputs entrassem nas tabelas correspondentes na base de dados principal (DTT), através da colocação de triggers nas mesmas.

A opção de ter um stored procedure, ou seja o cálculo todo na base de dados e não uma aplicação stand alone para fazer os cálculos, foi simplesmente porque assim a integração com a base de dados era mais eficiente e não estaria dependente de uma aplicação externa, podendo assim ser chamada a stored procedure só quando os inputs importantes caíssem na base de dados. Mas o ter uma aplicação stand alone ou a lógica integrada na base de dados, não trás vantagens ou desvantagens de nível significativo, tendo apenas sido uma opção de desenvolvimento.

Na figura 3.3 pode ser observado de um modo resumido as componentes e o processo envolvido num sistema de gestão de frotas.

Figura 3.3 – Ilustração dos componentes de um sistema de gestão de frotas



3.3.2.1 – optLOGIC

A lógica é chamada através de triggers colocados nas tabelas da base de dados principal que recebem os inputs dos veículos e as telemetrias, existindo um trigger para a tabela das mensagens pré-definidas, outro para a dos formulários e um último para as telemetrias.

O ponto de partida do algoritmo da lógica é uma main class que dependendo do tipo de input que recebe, irá chamar as classes e os seus respectivos métodos que irão tratar dos inputs, cada um da sua maneira, e inserir os mesmos na base de dados deste sistema.

O principal objectivo da lógica é garantir que a sequência de inputs definida é efectuada pelos motoristas. Uma vez que é um processo sequencial cada passo será validado no seguinte, ou seja só é possível saber se falta o passo n, quando se for inserir o passo n+1. Nos casos em que a lógica detecta que o passo n está em falta, o sistema vai inserir o input em falta de forma automática com valores predefinidos, que será a hora do registo correcto subtraindo 1segundo, enviando um alerta para a janela de monitorização de eventos da aplicação de gestão de frotas, ficando registado que foi feita uma inserção automática, que irá necessitar de posterior correcção.

Começando pelo passo inicial, o formulário de Inicio de Trabalho, quando a lógica recebe um formulário deste tipo vai proceder à verificação da coerência deste passo, vendo o contexto actual da sequência. Tem que verificar se existe algum trabalho aberto para o dia corrente, caso exista o input é descartado pois é uma repetição ou é considerado engano, isto porque só pode existir um trabalho por dia, logo não se considera caso não esteja fechado. Se estiver fechado, vai verificar se está no dia corrente ou no dia anterior, se for o próprio dia é também descartado pois só pode existir um trabalho por dia. Caso seja o dia anterior está tudo correcto e prossegue com as verificações. Se o trabalho estiver aberto mas

o dia de início for correspondente a outro dia anterior ao actual, o sistema vai fechar esse trabalho automaticamente e validar os dados para inserção do novo trabalho.

A validação da consistência dos dados do início de trabalho consiste em verificar os valores inseridos no formulário enviado pelo motorista. Caso algum dos dados não esteja correcto, é colocado um valor default definido com o cliente e sinalizado na interface de gestão de frotas um evento a informar que o existem dados que necessitam de correcção. Neste formulário em particular vai ser verificado se os km's inseridos são inferiores aos últimos enviados, se o armazém e o código do motorista inserido existem. Para este caso são só estes os dados pedidos no formulário que o motorista tem que preencher. Depois de verificados os dados do formulário é confirmada a integridade da lógica, ou seja validar se a mensagem enviada está na sequência correcta, caso não esteja quais os passos que estão em falta, ou se é uma repetição de mensagem.

Nesta classe está também o código que vai tratar da mensagem de fim de trabalho. Quando o sistema recebe um fim de trabalho vai verificar se o trabalho do dia corrente já foi fechado, caso se verifique então não insere e envia um alerta a informar este facto. Se estiver aberto mas o início de trabalho não corresponda ao dia actual, o sistema vai tentar inserir automaticamente um início de trabalho e de seguida faz o fim de trabalho. De notar que pode existir uma entrada só com início e fim de trabalho. Se tudo correr bem nestas verificações, então o sistema irá confirmar se existe alguma viagem em aberto,

isto pois segundo a sequência lógica o fim de trabalho só pode ser aceite se não houver nenhuma viagem por finalizar. Caso exista alguma viagem em aberto, o sistema vai tentar inserir automaticamente esse final de viagem e de seguida fecha o trabalho. Depois disto, ou caso tudo corra bem, o sistema fecha o trabalho e limpa todas as flags que possam existir activas, pois depois de se fechar o trabalho já não são necessários os temporizadores para alarmes.

De seguida vem a classe com os métodos para tratar das viagens, sendo que o primeiro método desta é aquele que vai validar o início de viagem.

Quando vai inserir um inicio de viagem a primeira verificação a ser feita é se o trabalho está fechado e se pertence ao próprio dia. Caso não seja do dia actual, então é considerado uma falha no fluxo da lógica e o sistema vai tentar inserir de modo automático o início de trabalho em falta e depois então insere o início de viagem correcto. No caso de estar fechado o trabalho do dia actual, então não insere a viagem e envia um alerta a informar que o trabalho já foi finalizado.

O outro caso a confirmar é se o trabalho está aberto mas também existe uma viagem em aberto. Neste caso a viagem é finalizada automaticamente e inserida a nova viagem. Esta solução trás o problema de que se o motorista se enganou e enviou um início de viagem quando não devia, vai criar outra a mais. No entanto tudo o que é inserido de modo automático é sinalizado e tem que ser rectificado por parte dos gestores de frota do cliente.

O método seguinte nesta classe é o que vai tratar o fim de viagem. Aqui vai ser validado se existe alguma viagem e se a mesma está em aberto, caso não esteja então é enviada uma mensagem a informar que não há viagens iniciadas. Se houver verifica se existe alguma carga, pois todas as viagens têm que ter pelo menos uma carga associada. Se não existir é informado este facto e inserida a carga automaticamente e só depois o fim de viagem. É verificado também se há alguma entrega a decorrer. Neste caso é então finalizada automaticamente a mesma e depois então pode ser fechada a viagem. Ao fechar uma viagem é conferido se o número de entregas efectuadas corresponde ao número de entregas previstas na guia de transporte associada a carga dessa viagem e essa guia de transporte é fechada.

Respectivamente à carga, temos os métodos de inicio de carga em que vai validar se os valores inseridos no formulário enviado são válidos, como por exemplo se existe alguma guia com o número enviado. Mais uma vez é verificada a recepção do input e a ultima mensagem recebida. No caso de não se encontrar na sequência são inseridos os valores para trás em falta, para que seja inserido esta mensagem na sequência correcta. O fim de carga não depende de uma acção humana directamente, esta é marcada quando o veículo sai do armazém, que está registado como um ponto de interesse (POI) e está em andamento. Isto é verificado pois devido aos veículos se encontrarem dentro de um edifício, os aparelhos perdem o sinal GPS, o que faz com que haja derivas, que visualmente se vai traduzir em entradas e saídas seguidas no POI.

Quanto ao restante das mensagens e formulários, como chegada ao destinatário, registo de operação, vão ter tratamento idêntico aos já mencionados, todos são verificados para certificar que são o passo certo para o fluxo da lógica e nos formulários são validados os campos para apanhar dados errados. O início de descarga é também uma telemetria, nomeadamente a abertura de portas, aliada ao facto de já ter feito chegada ao destinatário e o veículo se encontrar parado. Há no entanto três casos especiais a ter em conta, o caso das entregas, o fecho do trabalho e a reabertura de um trabalho.

Segundo o nosso cliente, um motorista pode fazer várias entregas no mesmo local de paragem, como por exemplo uma área que tenha mais do que um destinatário perto uns dos outros. Neste caso só se recebe uma chegada ao destinatário, mas o motorista terá que enviar tantos formulários de entrega quantas entregas foram efectuadas. Portanto o formulário de entrega é o único que permite a recepção de mais do que um seguido. Uma vez que o trabalho é o delimitador principal da actividade diária de um motorista, tem que ser garantido que não há fechos de trabalho indevidos. Inicialmente o que acontecia era que só se aceitava um fim de trabalho se a ultima mensagem recebida fosse um fim de viagem, caso contrário a mensagem de fim de trabalho seria descartada. Contudo havia um problema, o caso em que o motorista finaliza devidamente o trabalho, mas por algum motivo teria que reabrir o mesmo. Para contemplar este caso, o que foi feito foi que só no caso de receber um início de viagem depois de o trabalho estar fechado é que o mesmo seria reaberto,

qualquer outro input será descartado. Isto porque não podemos reabrir trabalhos caso se receba um outro input qualquer, pois isso poderia trazer erros ao sistema.

Estes são os mecanismos de regulação da lógica, contudo aliado a isto existe ainda um módulo de alarmes. Todos os alarmes gerados por introduções indevidas ou automáticas da lógica são enviados para a interface de gestão de frotas, havendo alguns que são enviados para o motorista. Mas há também alarmes ligados a temporizadores, como os alarmes de paragem ou os alarmes de temperatura.

Os alarmes de temperatura são os mais críticos. Estes são enviados para a interface, para o motorista e por email para os gestores de frota do cliente, com a informação do veículo, motorista, local onde o veículo se encontra, o tempo em que está fora da temperatura correcta e essa mesma temperatura.

Isto foi feito utilizando uma tabela de flags e um job, ou seja, sempre que um veículo pára no cliente ou indica que vai parar para almoço, é colocado uma flag com o timestamp desse momento. O job vai então correr sobre essa tabela e sempre que na coluna respectiva o intervalo de tempo pré-definido for ultrapassado, o alarme de tempo excedido é enviado. As temperaturas são configuradas na guia de transporte, sempre que uma guia de transporte é inserida com as temperaturas para um veículo, o aparelho desse mesmo veículo é programado com essas temperaturas para que assim que detecte que a temperatura está fora do intervalo, envie o sinal de alerta. O sistema vai então

receber esse sinal e colocar a flag activa e esperar o tempo pré-definido (15min) para então enviar o alarme. O tempo de paragem funciona do mesmo modo, só que neste caso é colocada a flag quando é feita a chegada ao destinatário ou o veículo é enviada a mensagem de sinal de paragem.

Esta foi a abordagem implementada para construir a lógica para responder às especificações do cliente

3.3.3 – Módulo de análise

O principal objectivo deste sistema é o armazenar dados de forma estruturada para poderem ser analisados posteriormente pelos administradores da empresa, de modo a conseguirem optimizar custos, tempo e terem uma noção real e detalhada de todos os factores de negócio da empresa.

Foram colocados à Com-UT uma série de templates de relatórios os quais deveriam ser disponibilizados, 11 na totalidade. Aliados a estes vieram alguns que existem para todos os clientes, como o relatório de viagens, onde é possível ver todas as viagens, com tempos de condução, paragem, distância percorrida por cada viagem efectuada de um veículo. Os relatórios pedidos tinham como informação tempos de carga, de descarga, tempo de espera no cliente e de espera no armazém, km's, temperaturas, períodos de trabalho de motoristas, abastecimentos, tempos de condução, tempos de paragem, médias e totais , entre outras informações. Todos podendo ser retirados por armazém, mostrando todos

os veículos afectos ao armazém seleccionado e por veículo ou motorista, no caso do relatório de períodos de trabalho de motorista.

A abordagem tomada para este módulo foi a criação de uma stored procedure em C#, que recebe os parâmetros de filtragem e o identificador do relatório e devolve os dados trabalhados, ou calculados para o relatório especificado. Tenho portanto uma stored procedure genérica, com classes contendo os algoritmos de extracção e cálculo dos dados de modo a representarem a análise pretendida para cada relatório.

Porquê utilizar uma stored procedure com as queries e os algoritmos em C# ao invés de fazer uma stored procedure para cada relatório, feita em transact sql (t-sql) no próprio sql server? O transact sql é uma extensão ao SQL mas que tal como este é bastante limitado na capacidade de se efectuar código que irá efectuar bastantes cálculos e mesmo queries muito complexas. Para ultrapassar esta limitação a Framework .NET dá-nos a possibilidade de integrar código CLR em C# que permite criar stored procedures, funções mais complexas e de um modo com melhor performance. Estas diferenças estão explicadas de um modo mais detalhado mais à frente no capítulo 4, secção 4.3.

A primeira razão tem a ver com portabilidade, mantendo uma stored procedure única que vai estar depois organizada por classes representando os relatórios, é mais fácil de gerir e é mais portável uma vez que não temos código espalhado por mais do que um local. A segunda razão e talvez a mais importante é a performance, em queries muito complexas o transact sql torna-se

muito pesado e lento, ao contrário do C#. Uma query mais complexa que leva quase 2 minutos a executar, corre em apenas 1 segundo quando implementada em C# e neste projecto todas as queries tiveram um nível de complexidade demasiado pesado para ser tratado pelo t-sql. Também sendo o C# uma linguagem de programação é mais fácil fazer algoritmos para efectuar os cálculos desejados do que em t-sql.

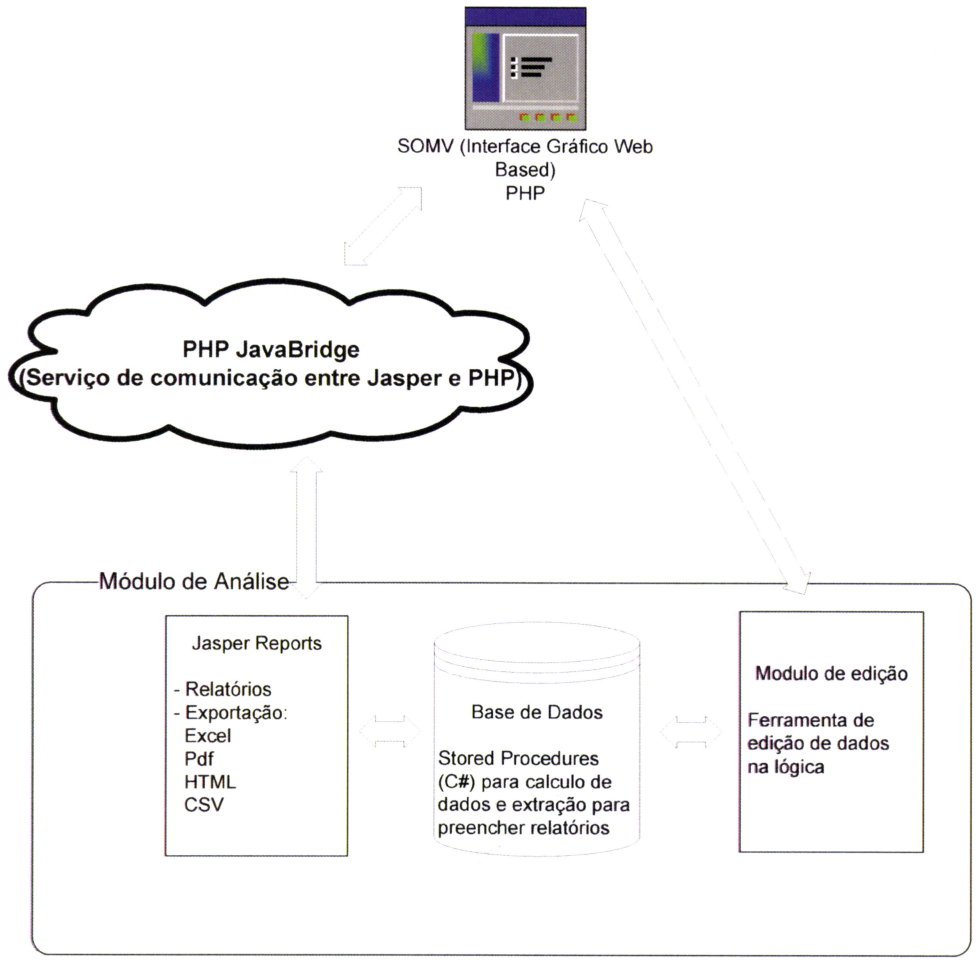
Por fim, a parte visual destes relatórios foi criada utilizando a ferramenta de reporting open source JasperReports. Com a utilização do iReport, uma ferramenta gráfica de construção de templates de relatórios, foram criadas todas as templates para os relatórios pretendidos. A utilização desta ferramenta foi escolhida pois havia a necessidade de ter uma opção de report que desse ao utilizador a possibilidade de seleccionar a exportação dos mesmos para ficheiros, pdf, excel ou csv e que desse para integrar com a aplicação de interface implementada em PHP. Neste ultimo ponto, foi necessário utilizar uma ferramenta para fazer a comunicação entre o PHP e o JasperReports que é totalmente feito em Java, sendo o mesmo um conjunto de bibliotecas Java. Esta comunicação foi efectuada utilizando a aplicação opersource PHPJavaBridge.

Depois disto este módulo foi integrado na aplicação de interface SOMV, já existente e disponibilizado ao cliente não como uma nova aplicação stand alone, mas como novas funcionalidades da ferramenta que já tinham acesso anteriormente.

Outra funcionalidade criada para este projecto, foi um modulo de edição de dados, em que o cliente através da selecção deste modulo no SOMV, poderia efectuar as alterações dos dados automáticos e ou errados.

Estas componentes podem ser visualizadas de um modo ilustrativo na figura 3.4.

Figura 3.4 – Modularização da componente de análise



3.4 – Aspectos de utilização

Nesta secção irei exemplificar o modo de utilização deste sistema, tendo no entanto omitido ou escondido qualquer informação que não tenha sido permitido publicar, quer pela empresa Com-UT, quer pelo cliente em questão.

O SOMV é uma aplicação web-based que permite ao utilizador efectuar a gestão da sua frota em tempo real e retirar relatórios para análise.

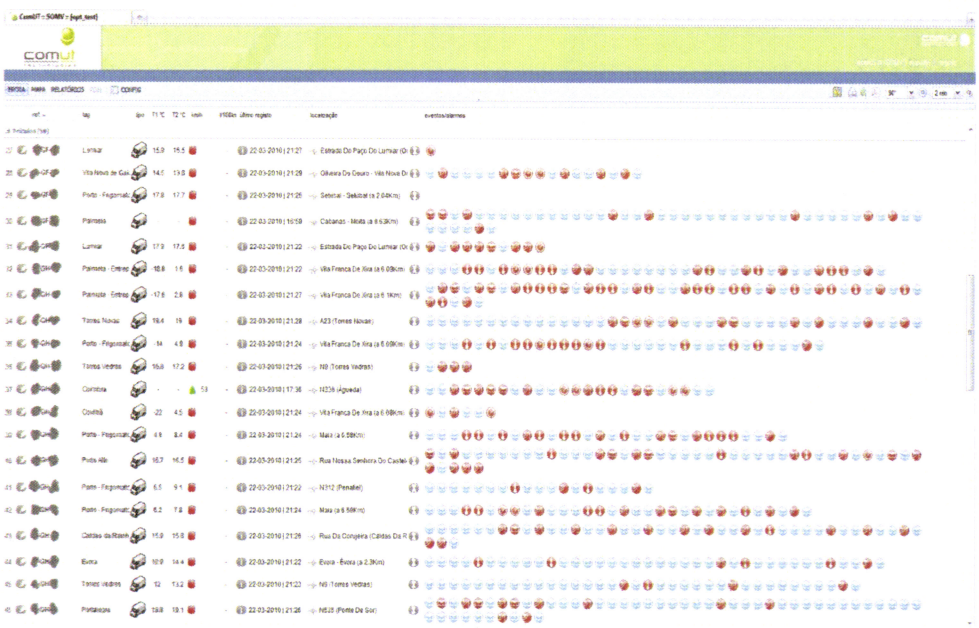
Após efectuar o login é apresentado ao utilizador a vista de listagem de eventos, onde o utilizador pode ver a lista por veículo com a informação onde o mesmo se encontra, um icone que informa se este está em andamento ou parado, a velocidade do mesmo em caso de andamento, o timestamp do último registo recebido, as temperaturas das sondas e os eventos programados para o user especifico. Poderão ser visualizadas outras informações bastando para tal escolher no header os campos que se querem visualizar ou retirar os que não se querem.

Aqui vai ser demonstrado o sistema configurado como para o cliente final da aplicação desenvolvida.

Portanto a informação visualizada por este cliente na parte dos eventos tem a ver com a sequência da lógica, onde são mostrado os inputs feitos pelos motoristas e também os alarmes despoletados pela lógica e ou outros alarmes por defeito, da aplicação.

Os eventos são visualizados na forma de ícones com duas cores, azul para os eventos que são registados de forma correcta e vermelho para os alarmes, alertas e inserções automáticas. Existe um conjunto de diverso de ícones, mas que podem ser partilhados por diferentes eventos. O utilizador passando com o rato por cima do ícone desejado irá ter uma tooltip com informação detalhada sobre a origem desse ícone, como poderá ser visualizado na figura 3.5 abaixo e clicando no ícone é possível deixar um comentário associado ao mesmo.

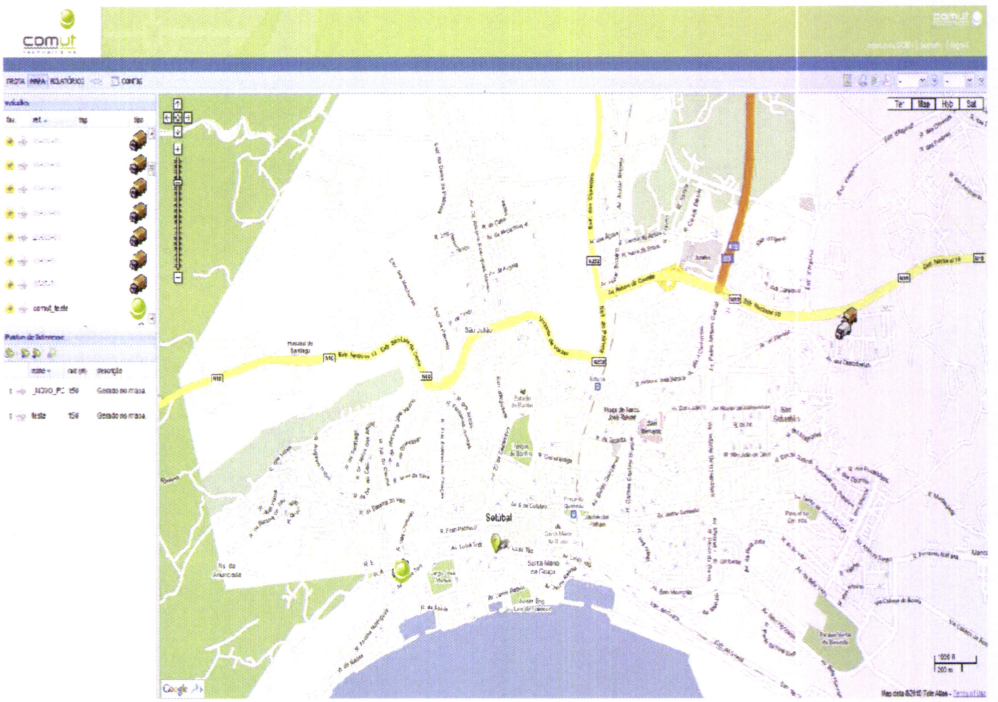
Figura 3.5 – Visualização de painel de monitorização de eventos



O utilizador tem também a possibilidade de visualizar o mapa com a posição dos veículos, vendo também no veículo um icone indicando se o

mesmo está parado ou se em andamento qual a direcção tomada. Nesta tab poderão ser também visualizados, criados ou apagados pontos de interesse ou POI's, que vem do inglês points of interest. Os mapas utilizados são mapas da Google, nomeadamente a sua API de Google maps, como se pode ver na figura 3.6.

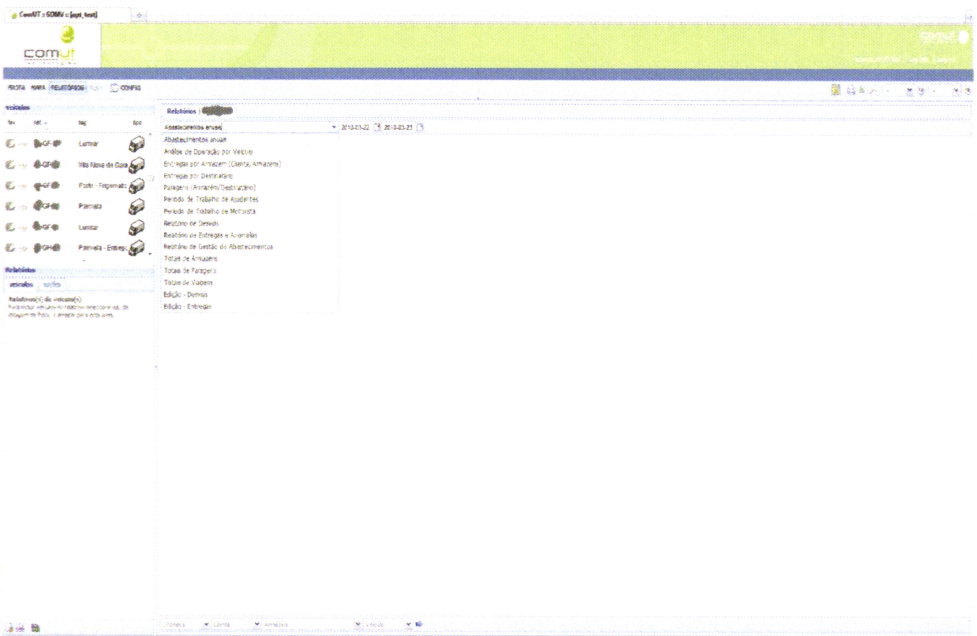
Figura 3.6 – Mapa de visualização de frota no terreno (Google maps)



A terceira janela de informação é referente ao menu de selecção de relatórios, portanto ao módulo de análise. Aqui o utilizador poderá escolher o relatório pretendido da lista com os relatórios definidos para este utilizador específico.

Os relatórios criados especificamente para este sistema de análise, estão agrupados num só tipo, ou seja o utilizador nessa listagem escolhe a opção "relatórios opt" e é depois visualizada uma janela onde o utilizador vai escolher ai o relatório correspondente a este módulo que pretende e de seguida os parâmetros de filtragem para o mesmo, sendo regra geral uma data inicial, uma data final, a empresa, o armazém e o veículo ou motorista. Um exemplo disto poderá ser visualizado na figura 3.7.

Figura 3.7 – Visualização do Painel de Escolha de Relatórios



Depois de escolhido o relatório é apresentado então a informação e o utilizador poderá depois exportar o mesmo para o formato pdf ou csv, através dos botões indicados para tal.

Existem ainda um outro relatório e um módulo de edição importantes para a lógica que pertencem à listagem inicial, que são eles o relatório de histórico de eventos, onde o utilizador pode seleccionar por veículo e por intervalo de datas e visualizar a listagem de eventos para esse veículo, pois essa lista na primeira janela só disponibiliza os eventos das últimas 24horas. O mapa de edição de lógica é uma ferramenta onde o utilizador poderá editar os dados automáticos e dados em falta ou mesmo dados enganados. Estes podem ser visualizados abaixo na figura 3.8 e figura 3.9 respectivamente.

Figura 3.8 – Mapa de Histórico de Eventos

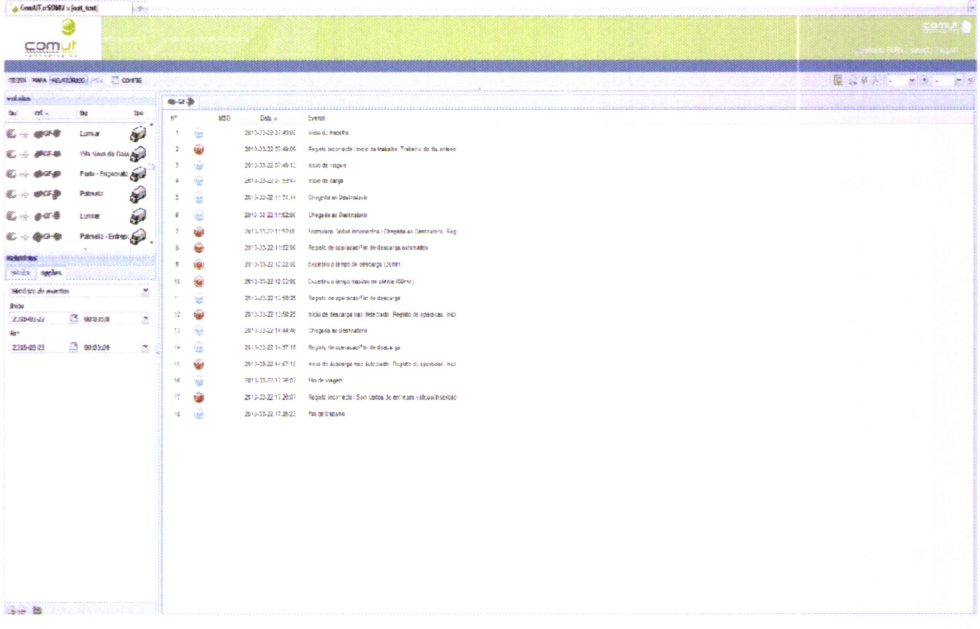
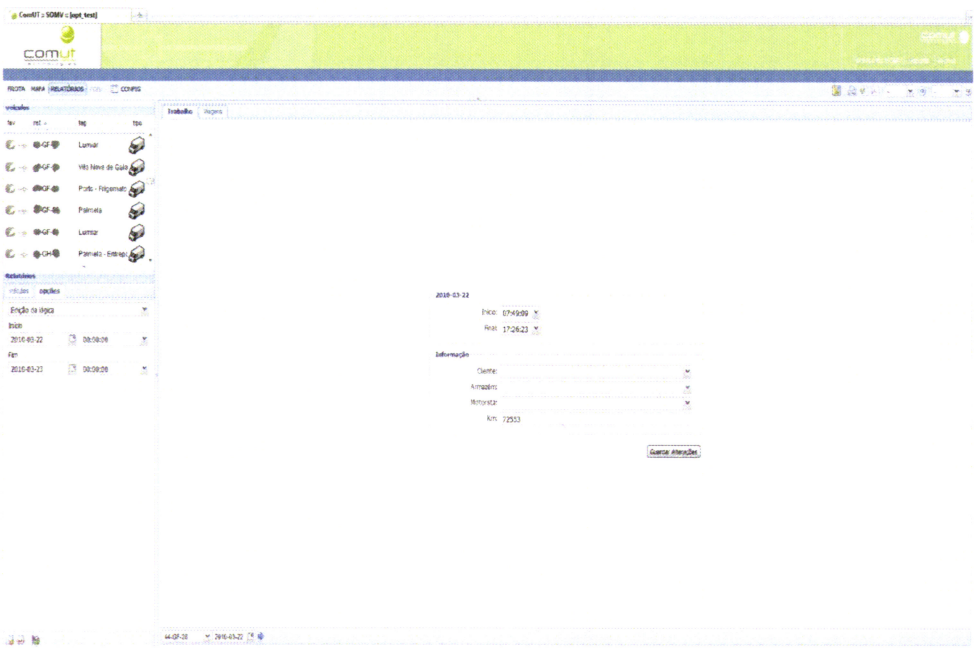


Figura 3.9 – Mapa de Edição da Lógica



Com isto termino a apresentação do projecto desenvolvido e o seu modo de utilização, passando à análise das ferramentas e metodologias utilizadas e também as possíveis opções para o mesmo efeito no próximo capitulo.

4 – Análise de ferramentas e metodologias

Este capítulo tem como objectivo evidenciar alguns resultados de pesquisa e análise sobre ferramentas e metodologias existentes no mercado, que poderiam ter sido utilizadas na implementação deste projecto. Esta análise terá como base o caso estudo desenvolvido e descrito anteriormente.

As análises efectuadas foram dirigidas no sentido do custo, performance e robustez das ferramentas ou arquitecturas de modo a tentar de um modo geral encontrar as vantagens e desvantagens de cada opção, pois não existe uma solução perfeita para todas as realidades. Isto porque existem muitos factores que influenciam uma escolha destas, há sistemas para os quais a fiabilidade e robustez é o mais importante, tal como dados de alto risco e alta disponibilidade, em que os custos não interessam. Por outro lado temos situações em que os custos são o factor mais importante, portanto à que tentar dar uma boa solução ao cliente, mas tentando reduzir ao máximo os custos para o mesmo.

Vou portanto passar à descrição dos resultados, falando separadamente de cada componente.

4.1 – Modelos de dados

4.1.1 – Base de dados relacional/transaccional vs Data warehouse

Nesta secção vou descrever alguns resultados da pesquisa e análise feita sobre estas duas metodologias de armazenamento de dados.

Não se pode abordar estes dois conceitos de um modo qualitativo, ou seja mostrar que um é melhor que o outro, pois estamos a falar de conceitos diferentes, com finalidades diferentes.

No entanto apesar de ambas estarem desenhadas e terem sido desenvolvidas para funções diferentes, a ideia aqui é analisar até que ponto é possível com a abordagem de uma base de dados relacional/transaccional atingir o máximo de objectivos de análise e reporting para o qual seria de esperar uma data warehouse.

Neste projecto foi utilizada uma base de dados relacional para gerir a chegada dos dados e para fazer a parte do módulo de análise.

Para começar será útil mencionar qual o princial objectivo de cada um dos modelos, as razões para as quais estas abordagens foram desenhadas.

A base de dados relacional consiste num modelo de dados que liga os dados utilizando características comuns entre eles. Deste modo faz ligações entre esses dados, mais propriamente, relaciona tabelas entre elas, de modo a garantir que cada tabela que se une a outra, partilham de características comuns entre elas.

O grupo de dados resultante destas uniões está organizado e é muito mais fácil de serem compreendidas.

Uma das características das bases de dados relacionais é a normalização dos dados. A normalização é um conjunto de boas práticas desenhadas para eliminar a duplicação de dados, que previne as anomalias de manipulação de dados e perda de integridade dos dados. Contudo a normalização é um processo que é criticado pois aumenta a complexidade e peso de processamento requerido para unir múltiplas tabelas que representam o que conceptualmente é um item singular.

A base de dados transaccional tem como principal funcionalidade ou tarefa, garantir a consistência e a integridade dos dados. Ou seja foi pensada para ter uma finalidade com mais ênfase no momento actual, a pedidos no momento e tem que garantir a funcionalidade e coerência dos dados em tempo real. Se durante uma transacção houver um problema no sistema que interrompa essa escrita, a base de dados transaccional, vai efectuar um rollback de modo a que a consistência e integridade dos dados na base de dados seja garantida. Como exemplo pensemos numa caixa multibanco, imaginemos que estamos a meio de um processo de levantamento de dinheiro e depois de efectuado o pedido ao banco acontece um problema e a máquina não chega a dar o dinheiro. Portanto houve uma falha no meio da transacção, logo o processo é revertido até que o estado da base de dados fique igual ao que estava no inicio da transacção. No exemplo, o valor que foi descontado da conta, será reposto, caso contrário

seria debitado esse valor à nossa conta e nunca tínhamos chegado a receber o mesmo em dinheiro.

Uma data warehouse por sua vez tem como conceito base que os dados guardados para análise tem um acesso mais eficiente se separarmos esses dados do sistema operacional. Ou seja uma data warehouse é uma colecção de dados que vai reunir informação de uma ou mais fontes, de maneira a conseguir construir uma nova base de dados central.

W. H. Immon [13] disse:

"A data warehouse is subject-oriented, integrated, time-variant and non-volatile [data] collection in support of management decision making processes"

Uma data warehouse é uma colecção de dados orientada a um tema, integrada, variável no tempo e não volátil para suporte aos processos de apoio à decisão.

Mais concretamente:

Orientada ao tema: todos os dados relevantes relacionados com um tema é reunida e guardada numa só base de dados.

Integrada: todos os dados na warehouse devem ser compatíveis uns com os outros, independentemente do tipo ou localização.

Variável no tempo: todos os dados contêm uma referência ao tempo de modo a que a idade de cada peça de dados possa ser determinada.

Não volátil: os dados não mudam a partir do momento em que foram recolhidos.

Por fim este tipo de abordagem tem as seguintes componentes:

- **Servidores de Base de dados:** são os locais onde os dados operacionais de um negócio estão acumulados, por norma em base de dados relacionais. É destes servidores que os dados que vão preencher a data warehouse vão ser recolhidos.
- **Queries/Relatórios:** Todos os processos que envolvem o pedido de dados a uma data warehouse para análise. Os relatórios são depois gerados para mostrar os resultados destas análises.
- **Data Mining:** o data mining é o processo de analisar os dados de negócio, para descobrir padrões e regras de informação, que podemos utilizar para melhor as operações do negócio. O data mining prediz as tendências futuras e comportamentos, permitindo às empresas tomarem decisões proactivas e baseadas em conhecimento.

Portanto temos um modelo de dados ou uma arquitectura de sistema que é mais vocacionado para análise e construção de relatórios, ou seja uma análise a longo prazo, podendo ser feita sobre dados do passado e já trabalhados e é uma

base de dados desnormalizada, pode existir repetição de dados. Isto dá-nos um poder de análise e eficácia muito superior, à que uma base de dados relacional, ou transaccional pode oferecer.

Observando e analisando estas características atrás evidenciadas e transpondo-as para a realidade dos sistemas de gestão de frotas a conclusão tomada é a de que claramente a opção ideal para estes sistemas será uma arquitectura de data warehouse. Estes sistemas têm duas componentes, a parte de recepção de dados e disponibilização desses dados em tempo real e a parte de análise e relatórios. Portanto teremos que ter os dois tipos de arquitectura, as bases de dados relacionais/transaccionais para a recolha dos dados que são enviados pelos veículos e que depois serão feitas queries para recolher esses dados e disponibilizar na interface para seguimento dos veículos e uma data warehouse que irá recolher os dados dessa ou dessas base de dados relacionais e que irá colocá-los numa data warehouse de acordo com a organização da empresa e da finalidade que se quer dar a essa informação, para que se possa retirar esses dados e efectuar uma análise mais poderosa, pormenorizada e eficaz, que será depois disponibilizada ao utilizador na forma de relatórios.

No sistema implementado a arquitectura de dados escolhida foi a utilização de base de dados relacional, normalizada para o armazenamento dos dados e sobre esse modelo de dados foi construído o módulo de análise. A melhor escolha teria sido a utilização de uma data warehouse para a recolha dos dados para análise, mas o cliente não aceitou a proposta devido ao investimento

que necessitaria de fazer e o tempo que teria que esperar para receber a aplicação. Significa portanto que a análise vai ser feita sobre a base de dados operacional em tempo real, ou seja é possível estarem utilizadores a retirar relatórios para análise ao mesmo tempo que o sistema está a receber dados dos veículos.

Na componente de gestão de frotas, em tempo real, que disponibiliza a informação dos eventos a decorrer e informações referentes ao veículo, não houve agravamentos pois este tipo de modelo de dados responde bem às necessidades existentes. Mas na componente de análise esta escolha trouxe algumas desvantagens. Quando se executa um relatório demasiado pesado, com cálculos mais complexos, o que acontece é que para retirar a informação para além de serem necessários fazer os cálculos da análise, a quantidade de joins de tabelas que é preciso fazer para retirar a informação causa uma grande carga de processamento e diminui assim a performance. O que acontece é que alguns tipos de filtragem levam bastante tempo a retornar os dados, aproximadamente 1min e outros dão timeout antes de sequer retornar os dados. Como por exemplo relatórios complexos em que o utilizador filtra por armazém, ou seja quer a mesma informação para todos os veículos afectos a um armazém, normalmente são entre 3 a 10 veículos. Se por armazém já é dado um timeout, por cliente muito menos retorna resultados, sendo que um cliente tem 1 ou mais armazéns. O nosso cliente tem ele mesmo vários clientes, cujos têm armazéns pertencentes, daí os clientes aqui mencionados referem-se aos clientes da

empresa que nos comprou o produto. Portanto, os ganhos que o cliente teria em ter aceite a opção de data warehouse teriam sido bastante superiores, mas nem sempre o factor performance e eficácia são os mais importantes.

4.2 – Servidor de base de dados

Nesta secção vou dar ênfase às tecnologias de servidores de base de dados. O objectivo aqui foi comparar as tecnologias com mais distribuição no mercado empresarial, sendo elas o Microsoft SQL Server 2008, ultima versão desta tecnologia, o Oracle 11g também a versão mais recente e como tecnologia open source o MySql, recentemente adquirida pela Oracle, mas mantendo uma politica diferente e PostgreSQL, que continua independente de qualquer empresa, contando com o contributo de uma comunidade de developers espalhada pelo mundo.

Mais uma vez não é objectivo desta análise escolher uma aplicação em detrimento de outra, porque tem funcionalidades que as outras não têm, mas sim evidenciar quais as principais vantagens e desvantagens de cada uma, pois com esse conhecimento se poderá dizer tendo em conta a realidade à qual a mesma vai ser empregue, qual a tecnologia que melhor se adequa.

4.2.1 – Microsoft SQL Server 2008

Esta plataforma tem oito versões disponíveis, as quais são descritas de uma forma resumida na tabela 4.1.

Tabela 4.1 – Versões de SQL Server 2008

Edição	Licenciamento	Preço de processador
Evaluation Edition	Disponível através de download na net	Grátis
Compact Edition	Disponível através de download na net; redistribuição permitida	Grátis
Express Edition	Disponível através de download na Web; redistribuição permitida	Grátis
Web Edition	SPLA, Licenciamento por Volume	**valor por processador por mês (SPLA) -** $3,500
Developer Edition	Retail, Licenciamento por Volume, OEM, MSDN	**Retail*** $50

Workgroup Edition	Retail, Licenciamento por Volume, OEM, ISVR, SPLA	**Retail*** $3,899
Standard Edition	Retail, Licenciamento por Volume, OEM, ISVR, SPLA	**Retail*** $5,999
Enterprise Edition	Retail, Licenciamento por Volume, OEM, ISVR, SPLA	**Retail*** $24,999

* Os valores apresentados são apenas para compras dentro dos Estados Unidos ou Canadá. Para as restantes partes do mundo estes valores são bastante superiores e dependem do redistribuidor e país.

Pontos Positivos:

- Preço das licenças mais baixo do que a sua concorrente Oracle 11g. O preço mantém-se o mesmo independentemente do número de cores que o processador tenha.
- O SQL Server inclui na sua versão Enterprise Edition tudo o que o seu concorrente Oracle 11g cobra como funcionalidades extra. Por exemplo a funcionalidade de spacial, que é útil para o negócio da gestão de frotas,

pois permite fazer o processo de geo-referênciação inversa, vem no SQL Server incorporado e para o Oracle é um extra que custa mais $3,800.

- Tem uma versão gratuita, a Express Edition, que é possível fazer o download da internet e com possibilidade de desenvolvimento para distribuição e venda, sem ter que pagar licença. Tem apenas a limitação de utilização de 1processador, 1GB de memória e um máximo de 4GB de espaço por cada data file. Por exemplo o software Primavera, ou o antivírus Panda Server edition, são exemplo da utilização comercial do SQL Server Express Edition.
- De acordo com a TPC (Transaction Processing Performance Council) o SQL Server 2008 tem os testes de benchmark mais aceitáveis para performance e relação preço/performance (benchmarks TPC-H e TPC-E). Apenas passado pela sua concorrente da Oracle no benchmark TPC-C.
- No campo da escalabilidade, o SQL Server 2008 trouxe algumas melhorias para dar suporte a sistemas de 64-bit com até 8Terabytes de memória.
- Alta disponibilidade através da possibilidade de instalação de várias instâncias passivas em servidores de baixo custo.
- O SQL Server 2008 oferece uma nova consola de administração de seu nome Resource Governor, que permite ao administrador de base de

dados (dba) configurar limites (especificar um mínimo de CPU, memoria RAM) e assignar prioridades no workload, como por exemplo quando duas instâncias coincidem num modo concorrente, assim não precisam interromper-se. Isto trás vantagens a nível de performance, pois deste modo temos tempo de resposta ao utilizador final consistentes.

- Na área da segurança o SQL Server 2008 é a plataforma mais robusta no campo dos sistemas de gestão de base de dados. Oferece o serviço built-in TDE (Transparent Data Encryption) que encripta e desencripta dados sem ter que acrescentar código na aplicação para o efeito. E esta funcionalidade já vem com o SQL Server 2008, ao contrário de Oracle que tem como extra. Trás também suporte EKM (Extensible Key Management) e HSM (Hardware Security Modules) que permitem a terceiros registar os seus próprios módulos no SQL Server de modo a fornecer gestão de chaves independentemente do motor do sistema de base de dados.

- Na área de Business Intelligence o SQL Server 2008 oferece uma componente de B.I. que acrescenta mais funcionalidades às já existentes capacidades de BI do SQL Server. Como o suporte para nível enterprise level data warehousing, online analytical processing (OLAP), reporting, scorecards, data mining, ETL, key performance indicators (KPI) e integração com Microsoft Office.

- Muitas empresas utilizam sistemas operativos Microsoft e outros produtos da marca, o que faz com que a escolha deste produto para integração com os já existentes na empresa seja mais provável.

Pontos Menos Positivos:

- Só é compatível com plataforma Microsoft Windows.
- Integração difícil de tecnologias não Microsoft com funcionalidades do SQL Server 2008, como por exemplo os Reporting Services. As funcionalidades agem bastante bem em conjunto quando todas são Microsoft. Mas quando se quer fazer integração de outras tecnologias, mostra-se um trabalho complicado e penoso.

Nota referente aos testes de benchmark TPC:

O TPC-C é um benchmark que simula um ambiente computacional complete, onde um conjunto de utilizadores efectua transacções contra a base de dados.

O benchmark TPC-H é um benchmark de suporte à decisão. Consiste num conjunto de queries ad-hoc orientadas ao negócio e modificações concorrentes de dados. Este benchmark ilustra sistemas de apoio à decisão, que examinam

grandes volumes de dados, executam queries com um grau de complexidade elevado e dá respostas a questões de negócio críticas.

Por último o benchmark TPC-E mede o número de transacções para obter resultados específicos do tipo de negócio, sustentável ao servidor durante um período de tempo. O benchmark é escalável, significa que o número de utilizadores a usar o sistema de teste pode variar para representar workloads de negócios com tamanhos diferentes.

Mais informações sobre estes benchmarks podem ser visualizadas no site do TPC (Transaction Processing Performance Council) [6].

4.2.2 – Oracle 11g

As versões disponíveis para este servidor de SQL são várias e com diferentes características, que podem ser observadas abaixo na tabela 4.2.

Tabela 4.2 – Edições de Oracle 11g

Edição	Licenciamento	Preço por processador
Lite Client		Gratuito
Lite Mobile Server	Retail, Licenciamento por Volume, OEM	$23,000
Xpress Edition	Distribuição livre	Gratuito
Personal Edition	Distribuição livre	Gratuito
Enterprise Edition	Retail, Licenciamento por Volume, OEM	$47,000
Standard Edition	Retail, Licenciamento por Volume, OEM	$17,500
Standard Edition One	Retail, Licenciamento por Volume, OEM	$5,800

Os valores aqui apresentados correspondem ao mercado dos Estados Unidos, podendo haver diferenças para outros mercados. No caso da Oracle o

cálculo de preços é mais complexo, porque tem que se ter em conta vários factores, como por exemplo os cores do processador, funcionalidades extra do servidor etc.

Pontos Positivos:

- Multi-plataforma: é possível utilizar em Microsoft Windows ou sistemas UNIX tais como IBM, Sun, Digital, HP, etc. Isto trás uma grande vantagem a nível de performance e fiabilidade, pois é possível instalar em sistemas mais robustos e fiáveis como o Linux. Ganha-se também em poder utilizar um sistema operativo livre de licenciamento, se bem que o preço das licenças de Oracle só por si já seja bastante elevado. O preço aumenta quanto mais cores o processador tiver.
- Versão gratuita a Oracle 11g XE (Xpress Edition). Esta versão tem limitações, sendo estas 4GB de limite de data file, 1GB de memória e 1 processador. Tem a possibilidade de ser utilizada em aplicações comerciais sem ser necessário o pagamento de nenhuma licença.
- Alta disponibilidade através da possibilidade de instalação de várias instâncias passivas em servidores de baixo custo.
- A consola de administração, nova no SQL Server 2008 (Resource Governor) já existe de um modo semelhante há algum tempo nos Servidores Oracle. No campo da administração de base de dados,

segundo um estudo efectuado pela empresa Edison Group, no Oracle 11g um DBA é mais rápido 44% a efectuar funções típicas de administração em comparação com o Microsoft SQL Server 2008. Poupa tempo ao necessitar de 43% menos passos para o mesmo conjunto de rotinas standard em relação ao seu concorrente da Microsoft, aumenta a produtividade e salva às empresas até $33,520 por ano, por DBA.

- A nível de segurança contem as mesmas funcionalidades que as referidas anteriormente para o MS SQL Server 2008, o que é uma vantagem. Contudo tal como já referido também, a desvantagem é estas funcionalidades terem que ser compradas como um extra, com um custo adicional de $10,000.00 por processador.
- Oracle Streams funcionalidade principal para replicação server-to-server. Suporta update-anywhere, contrariamente ao SQL Server, que tem suporte muito limitado para updates na réplica.
- Segundo o estudo comparativo feito pela Oracle, o seu sistema de replicação é mais robusto, fiável e tem melhor performance que o do seu concorrente SQL Server 2008. Isto faz com que a plataforma Oracle seja mais eficiente no campo de alta disponibilidade e sistemas de recuperação de desastres.

- A Oracle tem portanto vantagem no campo da replicação, clustering e alta-disponibilidade dos dados.

Pontos menos positivos

- Licenciamento demasiado alto, perdendo assim potenciais clientes, uma vez que existem opções à altura no mercado. Portanto clientes alvo bastante específicos, reduzindo o leque destes possíveis compradores.
- Funcionalidades extra cobradas à parte, quando o SQL Server oferece estas funcionalidades num só pacote a um preço mais reduzido.
- Recursos para Oracle mais difíceis de encontrar disponíveis na internet do que para o SQL Server 2008 [19], como por exemplo tutoriais em vídeo. Não é que a informação não esteja lá, mas é muito difícil de encontrar.
- Necessidade de gerir manualmente tablespaces (ou filegourps). No SQL Server 2008 estes crescem automaticamente. Apesar de não ser uma tarefa complicada, mas que poderá ter que ser feita várias vezes.
- Demasiado complexo, traduzindo-se num período de aprendizagem maior.
- Não tem indexes filtrados que permitem indexações num subconjunto de linhas de uma tabela. Estes indexes podem melhorar bastante a

performance em Data Warehouses. Permitem salvar espaço e optimizar a performance quando se insere ou actualiza conteúdos.

4.2.3 – MySql vs PostgreSQL

Coloquei o MySQL nesta análise apenas para referência a uma tecnologia Open Source. Isto pois analisando bem os pré-requisitos e as funcionalidades e capacidades oferecidas, não é um servidor que se enquadre nas necessidades do negócio.

É uma tecnologia útil e adequada para algumas aplicações que necessitem de um servidor de SQL para funcionalidades básicas, ou para finalidade de treino e estudo. Isto pois à parte a licença opensource em que diz que o código desenvolvido tem que ser aberto, o SQL Server e Oracle disponibilizam também versões gratuitas, com a vantagem de se poder distribuir comercialmente tudo o que se desenvolve sobre o mesmo.

O MySQL é um servidor de base de dados muito básico e depende em muito de uma camada exterior de programação para manipulação e preparação de dados, pode portanto pensar-se no MySQL mais como um local de armazenamento de dados do que como parte funcional de uma aplicação. Ao contrário de um servidor de base de dados como o SQL Server em que permite efectuar funções de negócio dentro do próprio servidor de base de dados. Como

stored procedures compiladas, funções definidas pelo utilizador, tipos de dados à medida, analysis services, reporting services, etc.

No campo do Open Source pode-se também ponderar outro servidor de base de dados que é o PostgreSQL. Inicialmente o MySQL foi desenvolvido com vista à velocidade enquanto o PostgreSQL foi desenvolvido focado nas características e standards. O PostgresSQL é um servidor de base de dados unificado com apenas um motor de armazenamento, enquanto o MySQL tem duas camadas, uma camada de SQL e um conjunto de 9 motores actualmente. Por exemplo os motores de MySQL mais utilizados são o InnoBD para suporte total ACID (ACID significa stands for *Atomicity*, *Consistency*, *Isolation* and *Durability)* e alta performance em grandes workloads com alta concorrência e o MyISAM para workloads de baixa concorrência. As aplicações podem combinar vários motores para retirarem as vantagens de cada um.

O PostgreSQL tem suporte para uma API assíncrona para utilização em aplicações que está reportado aumentar a performance em 40% em alguns casos [17]. O MySQL não tem esta capacidade assíncrona, apesar de alguns drivers terem sido criados para ultrapassar esta falha.

A nível de multi-processamento o MySQL historicamente tem-se focado mais na escalabilidade horizontal, ou seja a capacidade de conectar múltiplas entidades de hardware ou software de modo a funcionarem como uma unidade lógica singular. Já o PostgreSQL é muitas vezes considerado de escalar melhor com um número grande de cores ou níveis de concorrência pouco comuns.

Existem mais diferenças entre estes dois servidores que poderão influenciar a sua escolha num sistema que tenha requisitos que um servidor como estes dois possa ser compatível. Para mais informações poderá ser observado o site que está referenciado em [17].

Contudo o PostgreSQL contém uma capacidade para desenvolvimento em Business Intelligence que é única a este, que é o ASQL, que significa Analytic SQL Server que é a implementação do próximo modelo de Data Warehouse com capacidade para OLAP total e suporte para processamento analítico (modelos matemáticos, modelos estatísticos, etc) no servidor de SQL. A principal característica do ASQL é o desenvolvimento de soluções de B.I. (Business Intelligence) e A.I (Analytic Intelligence) de grande escala, não ligando às limitações conhecidas dos sistemas de hoje em dia, vastamente usada nas áreas do sector financeiro, banca, seguros, industria da saúde entre outros.

Poderá ser observada mais informação sobre o ASQL no site referenciado em [18].

Portanto a nível de Business Intelligence o PostgreSQL é a opção Open Source que poderá também ser ponderada juntamente com os concorrentes proprietários Microsoft SQL Server e Oracle.

4.3 – Linguagens de Programação

Neste capítulo vou analisar as linguagens utilizadas no SQL Server, pois foram aquelas com que trabalhei e poderei fazer uma análise mais concreta.

Portanto, as linguagens a analisar foram o C# (C – Sharp), T-Sql (Transact SQL)

4.3.1 – Transact Sql

O Transact-Sql ou T-Sql é uma extensão ao SQL e é da propriedade da Microsoft e Sybase. O T-SQL é a linguagem de programação nativa do SQL Server.

Esta extensão adiciona ao SQL algumas características, como controlo de fluxo, variáveis locais, varias funções com suporte para processamento de strings, matemática, processamento de datas, etc. e alterações aos statements UPDATE e DELETE.

Como a maioria das versões de SQL contém características de manipulação de dados e características de definição de dados. As características de manipulação de dados podem ser divididas nos seguintes grupos, uma linguagem de query declarativa (composta pelos statements SELECT/INSERT/UPDATE/DELETE) e uma linguagem procedimental (WHILE, assignment, triggers, cursores, etc).

O T-SQL no entanto é limitado tal como a sua base o SQL, stored procedures muito complexas feitas em T-SQL, acabam por ter tempos de resposta demasiado elevados e por isso vão criar uma quebra de performance quando são utilizadas. Para dar a volta a situações como esta e outras, a Microsoft integrou o C# no SQL Server. O C# não vem integrado em si no SQL Server, mas através da utilização do Visual Studio, é possível criar por exemplo stored procedures e fazer o deploy das mesmas na base de dados alojada no SQL Server. Aconteceu por exemplo situações em que uma stored procedure desenvolvida em T-SQL demorava aproximadamente 2minutos a retornar os valores, em C# a mesma stored procedure devolvia o resultado em aproximadamente 1segundo.

4.3.2 – C# (C Sharp)

Esta é uma linguagem criada pela Microsoft e desenvolvida para a sua plataforma. NET. É uma linguagem orientada a objectos baseada na linguagem C++, mas tem também algumas bases em Java e Delphi.

Algumas das características desta linguagem são o suporte de apontadores, garbage collector, não existe herança múltipla, uma classe de C# herda apenas de uma só classe, podendo apenas ser simulado a utilização de herança múltipla através de interfaces, entre outras.

O C# é uma linguagem CLR (Common Language Runtime) que permite aos programadores desenvolverem procedures, triggers e funções e integra-los na base de dados. Isto permite também aos programadores adicionarem à base de dados novos tipos e agregados.

As linguagens CLR providenciam uma alternativa à parte procedimental do SQL, já falado na secção do T-SQL. Contudo é importante realçar que as aplicações da base de dados devem utilizar a linguagem de query declarativa o máximo possível. Esta parte da linguagem é capaz de aumentar o poder do processador de query, que tem melhores capacidades de optimizar e executar operações em massa. Portanto deve-se apenas utilizar linguagens procedimentais para expressar lógica que não pode ser expressada dentro de uma linguagem de query.

Deve-se então utilizar o máximo possível Transact-SQL, os programadores devem ver o CLR como uma boa alternativa quando não é possível exprimir a lógica na linguagem de query.

Resumindo e citando a Microsoft [15]:

Algumas recomendações que podem ser utilizadas quando se escolhe entre utilizar CLR ou T-SQL:

- Utilizar os statements do T-SQL, SELECT, INSERT, UPDATE e DELETE sempre que possível. Processamento procedimental e row-

based devem ser utilizados apenas quando a lógica não dá para expresser utilizando a linguagem declarativa.

- Se o procedimento for simplesmente um wrapper para comandos declarativos de T-SQL, então deve ser escrito em T-SQL.
- Se o procedimento envolver em primeira instância navegação por coluna forward-only e read-only através de um conjunto de resultados com algum processamento de cada coluna, utilizar o CLR é provavelmente mais eficiente.
- Se o procedimento envolver acesso a dados significante e computação, considerar separar o código procedural em uma parte CLR que chama um procedimento T-SQL para efectuar o acesso aos dados, ou um procedimento T-SQL que chama um CLR para efectuar a computação.

Portanto a escolha desta linguagem para o desenvolvimento, teve como principal razão a sua grande versatilidade, performance e o facto de se poder integrar com o SQL Server 2008 através do Visual Studio para o desenvolvimento de stored procedures, que são integradas na base de dados.

Pois foi necessário efectuar cálculos que não eram possíveis de expressar numa linguagem de query, nomeadamente o T-SQL. Esta capacidade do C# trouxe uma grande vantagem aos programadores e também à própria plataforma.

Todo o sistema da aplicação, os algoritmos de lógica e os algoritmos de reporting foram desenvolvidos nesta linguagem, tirando partido das suas potencialidades e ficando assim integrado na base de dados como uma stored procedure que é chamada através de triggers.

4.4 – Ferramentas de reporting

Muitas são as ferramentas disponíveis para reporting, aqui a análise vai recair sobre duas ferramentas que foram utilizadas e ou estão a ser utilizadas no momento para efectuar a parte dos reports, que são elas o JasperReport e os Reporting Services da Microsoft.

4.4.1 – JasperReport

O JasperReport foi a tecnologia utilizada para implementar a parte de reporting deste sistema. Esta foi a escolha feita, pois é uma tecnologia opensource que é facilmente integrável com a aplicação existente, desenvolvida em php.

Uma vez que esta tecnologia é desenvolvida em Java, foi necessário utilizar uma ferramenta de seu nome PHP JavaBridge, também open source, que cria uma serviço que faz de interface entre o PHP e o Java, de modo a ser possível chamar os métodos da API do JasperReports através do PHP. Uma das

desvantagens desta solução é a instabilidade deste serviço, pois tem quebras mais frequentes do que se deseja num sistema deste tipo.

O JasperReport é uma plataforma de reporting totalmente escrita em JAVA e utiliza dados vindos de fontes de dados variadas para produzir relatórios que podem ser visualizados, impressos ou exportados em vários formatos como PDF, HTML, Excel, Word e OpenOffice. É gratuito, tendo também uma versão paga que é o JasperServer Pro, que trás como extra a possibilidade de criação de relatórios AD-HOC e dashboards, é um servidor onde é possível guardar modelos de relatórios, relatórios e disponibilizá-los no lado do servidor e até agendar tarefas de criação de relatórios ou envio destes para emails definidos.

Para o objectivo deste projecto, foi utilizada a versão gratuita pois só era necessário o JasperReport para a criação de templates de relatórios e exportação para formatos como Excel, CSV e PDF.

O Jasper tem também uma ferramenta de desenho de templates de relatórios, nomeadamente o iReport, que é um ide que permite uma construção visual do relatório, através de drag and drop de objectos, com uma utilização bastante intuitiva e acessível. É fácil de utilizar tendo alguns pontos menos positivos, como a biblioteca de gráficos com uma apresentação bastante fraca, a necessidade de utilização de código java através de scriptlets para fazer cálculos ou mesmo configurações de objectos mais complexos, uma vez que a utilização de código java directamente nas propriedades do objecto é limitado, não dando

para fazer blocos de instruções. Como por exemplo, não tem capacidade para tratamento directo de objectos do tipo datetime. Para calcular diferenças de tempos, é necessário passar o valor para segundos em inteiro, efectuar os cálculos e passar as partes do tempo, hora, minutos e segundos como string e concatenar para obter no final a diferença de tempo e apresentar no relatório. Mas no geral é uma ferramenta bastante versátil, de fácil utilização e capaz de responder a grande parte das necessidades pedidas a uma plataforma destas. Mas nada comparado com as funcionalidades e capacidades dos Reporting Services da Microsoft, contudo a parte de construção do layout é melhor no iReport que no ReportDesigner da Microsoft, isto pela minha experiência de utilização dos dois. Esta ferramenta divide o relatório por bandas, como o Page Header, Page Footer, Group Header, Group Footer, Detail, Background, etc. O designer da Microsoft tem o header o footer e o body, o iReport está estruturado de uma maneira mais user friendly e intuitiva.

4.4.2 – MS SQL Server Reporting Services

Os Reporting Services são a plataforma de reporting da Microsoft que vem integrada com o SQL Server, nas versões mais completas.

A vantagem dos Reporting Services é a sua capacidade de trabalhar os dados, de modelação dos mesmos e apresentação. Uma novidade dos Reporting Services 2008 é a introdução das tablix, que é a conjugação das capacidades de

uma tabela e de uma matrix. Com isto é possivel fazer relatórios com informação mais detalhada e completa. Por exemplo é possível ter um relatório de vendas agrupado por linhas e por colunas como exemplificado na figura 4.1.

Figura 4.1 – Exemplo de um relatório utilizando tablix (retirado do site: http://prologika.com/CS/blogs/blog/archive/2007/08/12/tablix-the-crown-jewel.aspx)

			By Year					
			2002				2003	Pacifi
	Q1	Q2	Q3	Q4	Total		Total	Total
⊞ Bikes	$5,869,305	$5,895,800	$8,072,160	$7,027,269	$333,387		$351,983	!
⊟ Components	$175,044	$376,247	$1,935,906	$1,123,845	$30,345		$17,318	
Bottom Brackets							$30,793	
Brakes							$45,231	
Chains							$5,694	
Cranksets							$124,371	
Derailleurs							$44,359	
Forks			$26,167	$23,543	$49,710		$28,259	
Handlebars			$35,341	$18,309	$53,651		$88,769	
Headsets			$19,702	$16,382	$36,084		$25,852	

Podemos ver então a capacidade de uma tablix, ganhamos a flexibilidade de uma tabela e as capacidades de crosstab de uma matriz.

O servidor dá-nos a possibilidade de aceder aos relatórios através de acesso por URL. Este acesso vai abrir o reportviewer do próprio reporting service e mostrar o relatório utilizando as capacidades interactivas deste serviço, portanto deixamos de ter apenas um relatório estático que apenas podemos visualizar e exportar ou imprimir. Por exemplo, podemos criar um relatório e dizer que queremos que seja possível expandir e colapsar por grupo, para melhor visualização dos agrupamentos ou por exemplo através de um elemento

do relatório indexar a outro ou outros relatórios, criando um link entre si. Há também a possibilidade de utilizar o webservice para acesso através da API.

Contudo e apesar destas funcionalidades, a integração dos Reporting Services com aplicações externas, desenvolvidas por terceiros, demonstra-se um obstáculo trabalhoso de se ultrapassar. Isto porque as aplicações Microsoft foram pensadas mais para a sua utilização de integração com outras aplicações da mesma casa e não com aplicações externas. As ajudas na internet para ultrapassar este problemas também não são muitas e os canais da própria Microsoft, apesar de terem uma solução, esta vem com bugs reconhecidos pela equipa técnica nos fóruns. Portanto, ganha-se em capacidade de tratamento dos dados, em modularização, mas perde-se em simplicidade de desenho de layout e de integração com o sistema externo. Contudo e tendo em conta as vantagens que esta ferramenta pode trazer, a transição do Jasper Reports para os Reporting Services é o caminho que está a ser seguido.

A ferramenta de construção de layout de relatórios é o ReportBuilder, que pode ser standalone, ou usando o Visual Studio. Esta ferramenta em comparação com o iReport é menos intuitiva, menos user friendly, mais complicada e com um custo de tempo/relatório maior. Por exemplo ao contrário do iReport, que se podia seleccionar vários objectos e modificar parâmetros comuns a todos, no ReportBuilder só se consegue aceder às propriedades de apenas um objecto de cada vez. A estrutura das várias regiões é menor que no

iReport, tornando as coisas menos intuitivas. Contudo a capacidade de colocar várias linhas de código no template é superior ao iReport.

Resumindo, ambas as aplicações têm os seus fortes e os seus pontos menos bons, no entanto se fosse possível juntar numa só aplicação os pontos positivos de cada uma teríamos uma ferramenta quase perfeita.

5 – Conclusões e perspectivas futuras

O principal objectivo deste trabalho foi desenvolver um sistema que respondesse às necessidades especificadas pelo nosso cliente e também fazer uma análise às ferramentas e metodologias utilizadas no decorrer do mesmo.

O sistema foi desenvolvido e está actualmente em produção, sendo utilizado pelo cliente diariamente. Como primeiro sistema que foi, continua em fase de optimizações, mas o núcleo mantém-se o mesmo.

O cliente pode seguir os eventos dos motoristas na interface, observando os eventos feitos correctamente e vendo os alertas para aqueles que são automáticos. Tem também uma funcionalidade de edição de dados, para alterar os dados inseridos automaticamente, um relatório onde pode ver a sequência dos eventos por viatura, mas com o período desejado, uma vez que no ecrã de monitorização de frota só aparecem os eventos das últimas 24horas.

Tem também o módulo de análise com todos os relatórios pedidos, com o layout e informação especificada no início do projecto.

Uma das conclusões visíveis deste sistema foi uma falha que tinha sido prevista e que acabou por acontecer. Como descrito, inicialmente o sistema foi pensado para parar caso surgir-se um input fora de sequência, obrigando o motorista a efectuar primeiro o input em falta e de seguida o que queria fazer. Mas a pedido do cliente, a lógica não podia falhar, inserindo automaticamente valores onde os mesmos faltassem, havendo o compromisso da parte dos

utilizadores que esses dados automáticos seriam corrigidos. Contudo, essa falha esperada aconteceu, os dados não são corrigidos o que criou alguma inconsistência nos resultados do módulo de análise. Este é um ponto a melhorar no futuro, criar mecanismos de detecção automática de falha de envio de input, para que o valor inserido seja mais próximo do valor real. Isto poderá ser efectuado utilizando um motor de regras e ficando à escuta de telemetrias pré-definidas, como saída de POI, velocidade superior a 0, etc.

Outro ponto que não foi esperado, foi o facto de o retorno dos relatórios quando pedido um número muito grande de dados, ou seja várias viaturas, com dados a efectuar cálculos bastante pesados, o processo rebenta antes de retornado os valores. Aqui o melhoramento futuro só passará pela criação de uma data warehouse para o módulo de reporting, para que seja possível ter os dados calculados guardados, tornando o acesso aos mesmos mais rápido, melhorando assim a performance.

Mais um melhoramento futuro poderá passar pela utilização de um motor de eventos, para fazer um controlo mais estendido dos eventos diários da lógica. Poderá ser utilizando o motor de eventos do SQL Server 2008 RC2, StreamInsight, o Esper ou Nesper que são ferramentas Open Source, sendo que uma é baseada em Java e outra em C# respectivamente, ou outra idêntica, aquela que melhor se enquadre na realidade das necessidades do projecto.

A análise feita para este projecto ajudou-me também a ganhar algum conhecimento extra e que irá ser útil em futuros desenvolvimentos, acerca das ferramentas utilizadas e metodologias.

Durante o desenvolvimento deste projecto algumas escolhas tiveram que ser feitas e outras que não tiveram opção de escolha. Contudo todas as escolhas feitas foram ponderadas e consideradas as que mais se enquadravam e respondiam às necessidades que apareceram no momento. Não pondo de parte qualquer alteração às mesmas, caso no futuro se apresente uma solução melhor ou que seja possível utilizar e que no momento do desenvolvimento deste projecto não foi possível.

Com isto concluo a descrição do sistema desenvolvido e da análise feita sobre o mesmo.

Referências

[1] Ryberg, T. 2009. *Fleet Management and Wireless M2M* Berg Insight, Sweden

[2] Wikipedia.org. 2010. *Fleet Management* [online]. [Acedido: 21 de Março de 2010]. Disponível em:

http://en.wikipedia.org/wiki/Fleet_management

[3] Wikipedia.org. 2010. *Oracle Database* [online]. [Acedido: 21 de Março de 2010]. Disponível em:

http://en.wikipedia.org/wiki/Oracle_Database

[4] Oracle. Oracle Corporation. 2008. *Technical Comparison of Oracle Database 11g versus SQL Server 2008: Focus on Replication.* [online]. [Acedido: 21 de Março de 2010]. Disponível em:

http://www.oracle.com/technology/deploy/availability/pdf/cwp_replication_oracle11gr1_ss2008.pdf

[5] Oracle. Oracle Corporation. 2008. *Oracle Real Application Clusters 11g. Technical Comparison with Microsoft SQL Server 2008.* [online]. [Acedido: 21 de Março de 2010]. Disponível em:

http://www.oracle.com/technology/products/database/clustering/pdf/twp_racsql server_2008.pdf

[6] TPC.org. 2010. *TPC Benchmarks.* [online]. [Acedido: 21 de Março de 2010]. Disponível em:

http://www.tpc.org/tpcc/default.asp

[7] Das, V. 2008. *Oracle 11g vs. Microsoft SQL Server 2005/2008.* [online]. [Acedido em: 21 de Março de 2010]. Disponível em:

http://oracleappstechnology.blogspot.com/2008/06/oracle-11g-vs-microsoft-sql-server-2008.html

[8] Olamendy, J. 2009. *Oracle database 11g and Microsoft SQL Server 2008.* [online]. [Acedido em: 21 de Março de 2010]. Disponível em:

http://www.c-sharpcorner.com/UploadFile/john_charles/Oracledatabase11gandMicrosoftSQL

Server20080821200914245 6PM/Oracledatabase11gandMicrosoftSQLServer20 08.aspx

[9] Microsoft. Microsoft. 2009. *SQL Server compared to Oracle.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://www.microsoft.com/sqlserver/2008/en/us/compare-oracle.aspx

[10] Oracle. Oracle corporation. 2010. *Oracle Technology Global Price List.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://www.oracle.com/corporate/pricing/technology-price-list.pdf

[11] Microsoft. Microsoft. 2010. *Preço do SQL Server 2008.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://www.microsoft.com/brasil/servidores/sql/howtobuy/pricing.mspx

[12] MWolk blog. 2009. *Transactional Database.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://mwolk.com/blog/transactional-database/

[13] PharmacyInformatics. PharmacyInformatics.org 2009. *Data Warehouse.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://www.pharmacyinformatics.org/datawarehouse.htm

[14] Wikipedia.org. 2010. *Transact-SQL.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://en.wikipedia.org/wiki/Transact-SQL

[15] Microsoft. Microsoft Corporation 2010. *Using CLR Integration in SQL Server 2005.* [online]. [Acedido em 21 de Março de 2010]. Disponível em:

http://msdn.microsoft.com/en-us/library/ms345136%28SQL.90%29.aspx

[16] Experts Exchange. *Pros and Cons of SQL SERVER vs MySQL...which should I choose.* [online]. [Acedido em 27 de Abril de 2010]. Disponível em:

http://www.experts-exchange.com/Microsoft/Development/MS-SQL-Server/Q_22973951.html

[17] WikiVS. 2010. *MySQL vs PostgreSQL.* [online]. [Acedido em 27 de Abril de 2010]. Disponível em:

http://www.wikivs.com/wiki/MySQL_vs_PostgreSQL

[18] PostgreSQL. 2008. *Analytic SQL Server – next generation analytic Data Warehouse with OLAP support.* [online]. [Acedido em 27 de Abril de 2010]. Disponível em:

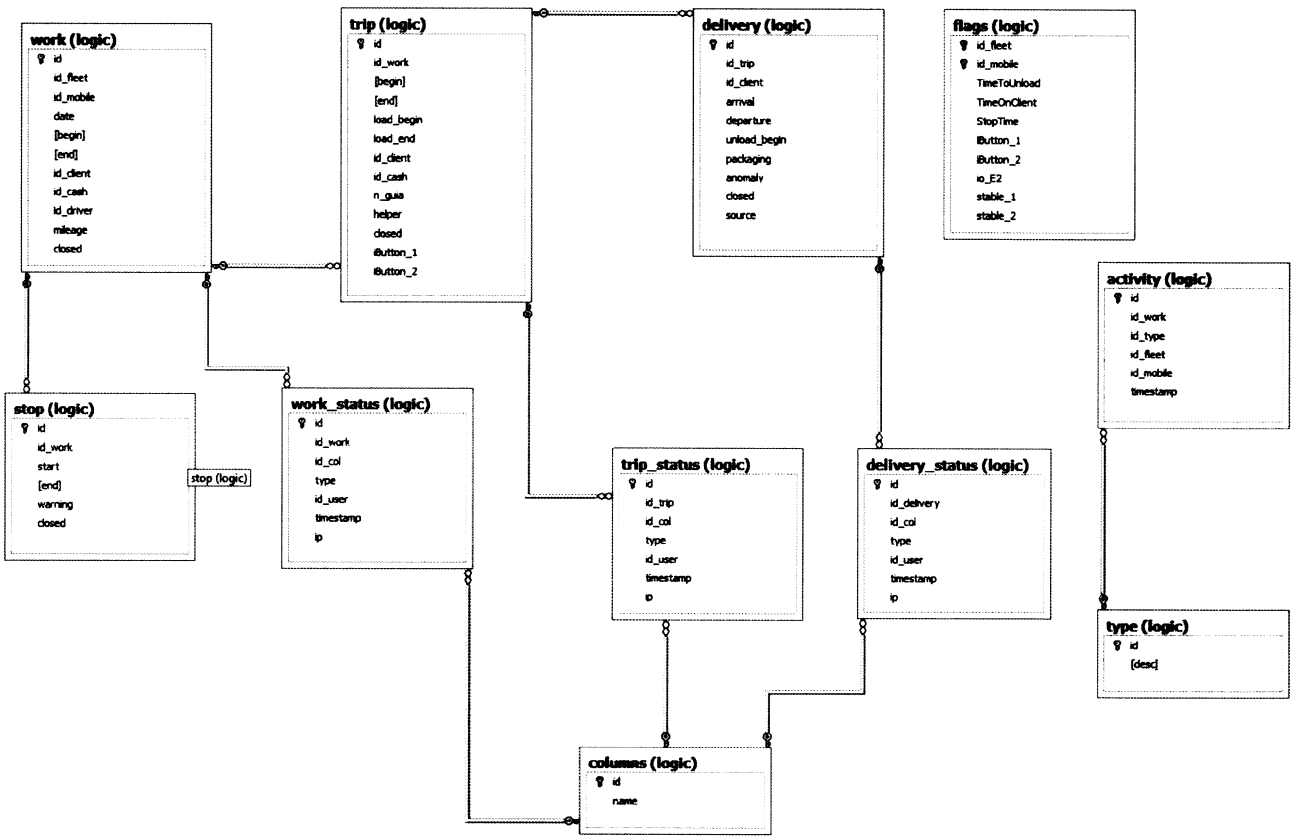
http://www.postgresql.org/about/news.909

[19] InfoWorld. MCCOWN, S. 2008. *Oracle vs SQL Server.* [online]. [Acedido em: 27 de Abril de 2010]. Disponível em:

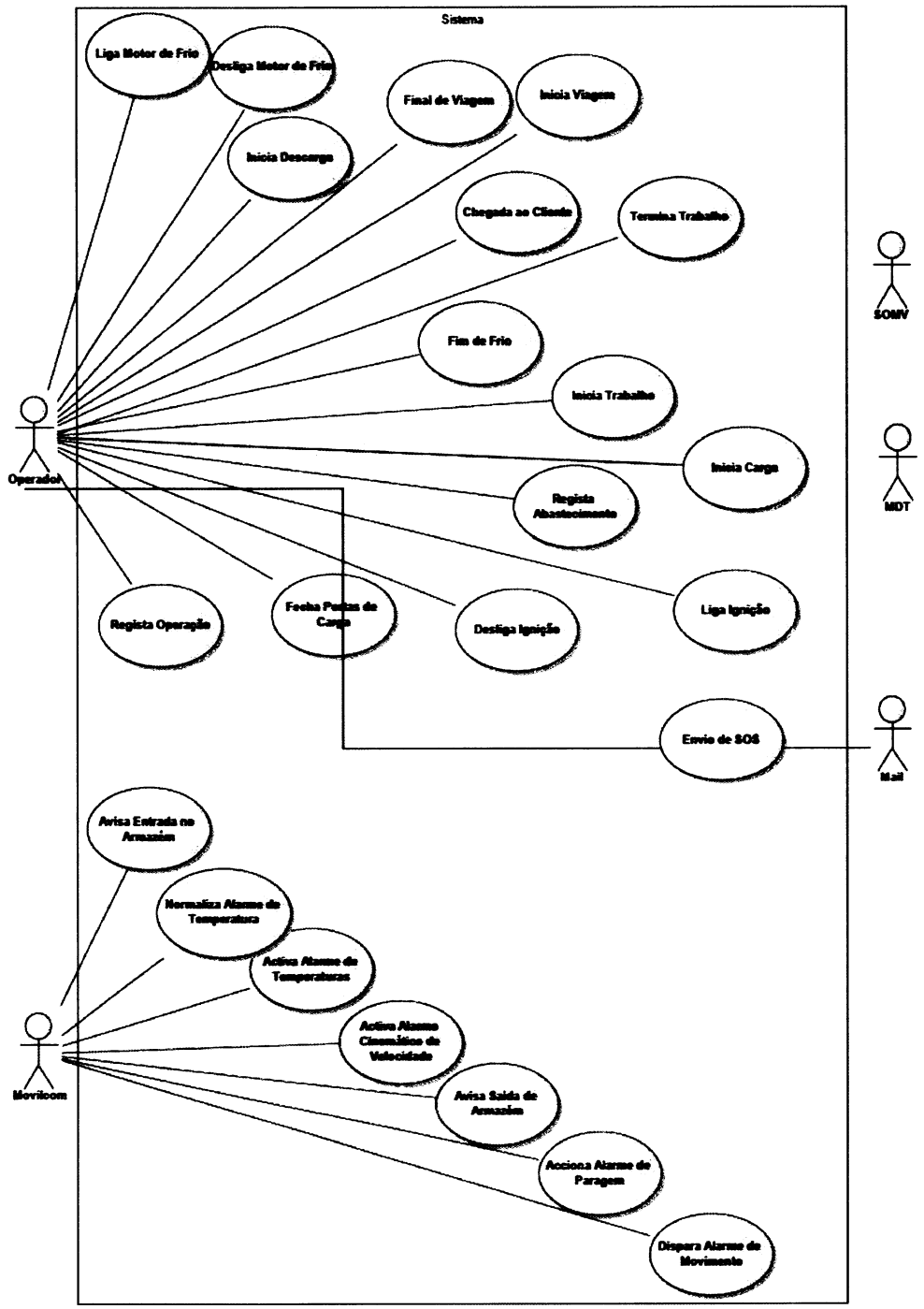
http://www.infoworld.com/d/data-management/oracle-vs-sql-server-758?page=0,1

ANEXOS

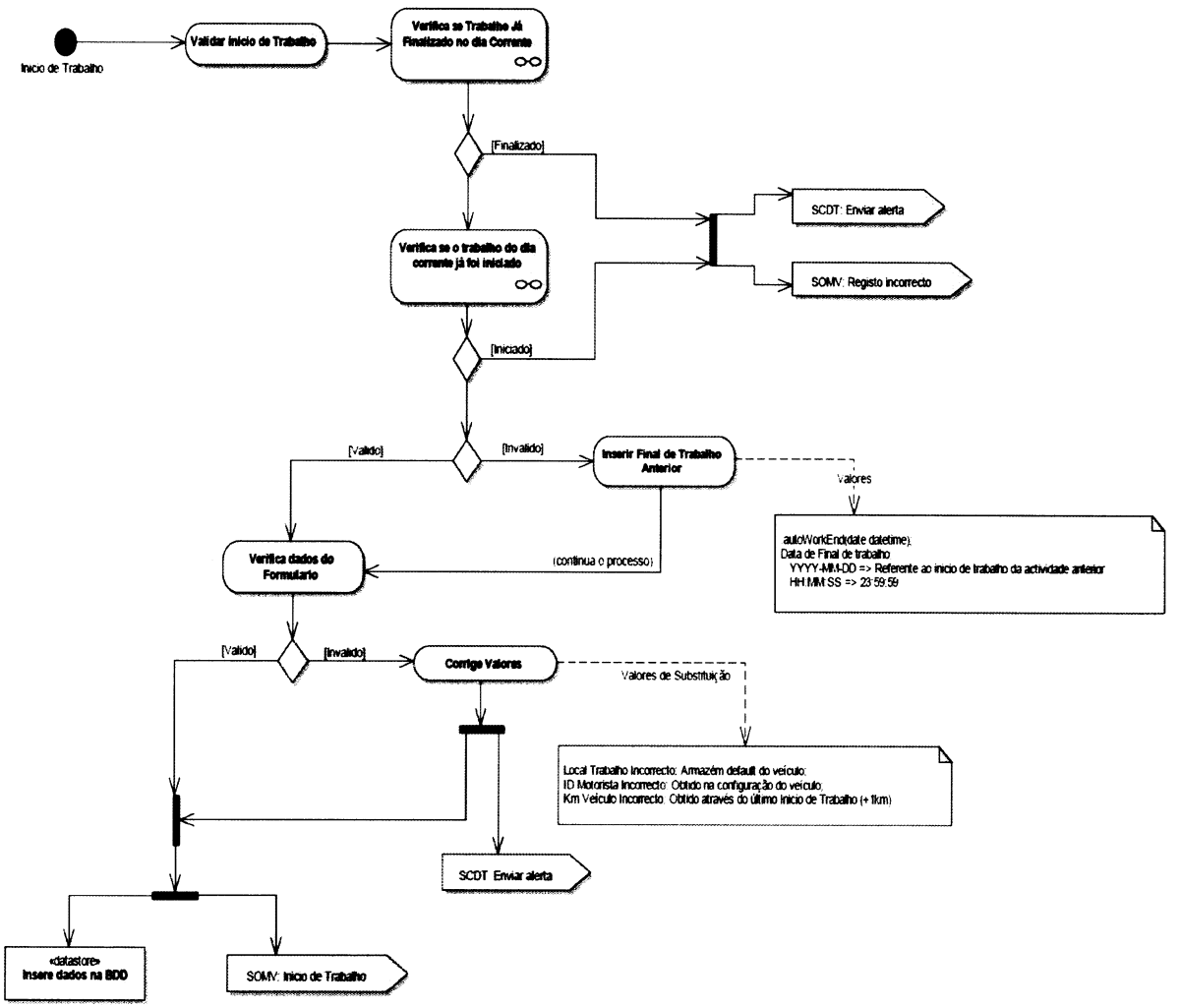
Anexo A – Modelo de dados



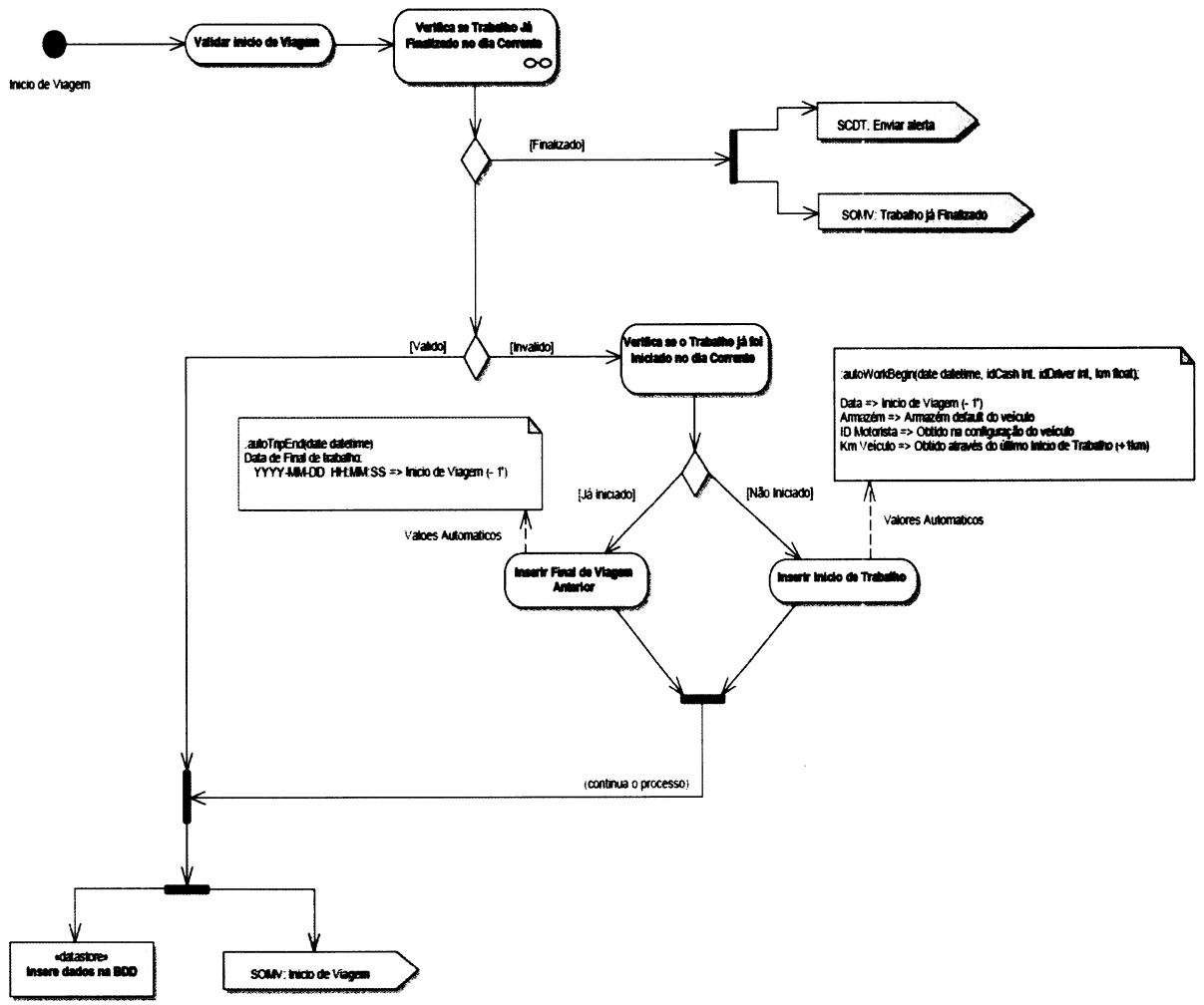
Anexo B – Diagrama de Use Cases



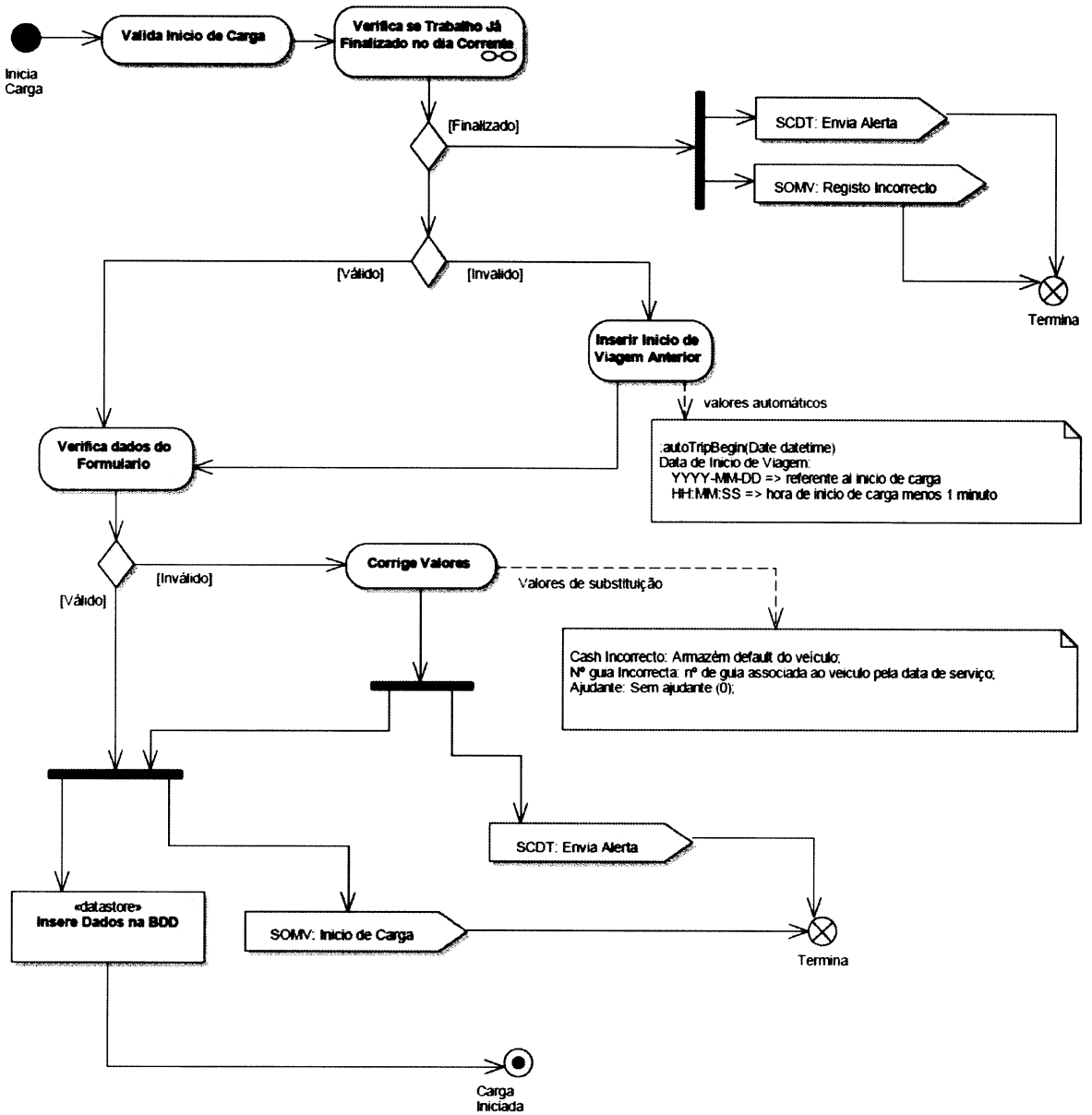
Anexo C – Diagrama de Actividade de "Inicio de Trabalho"



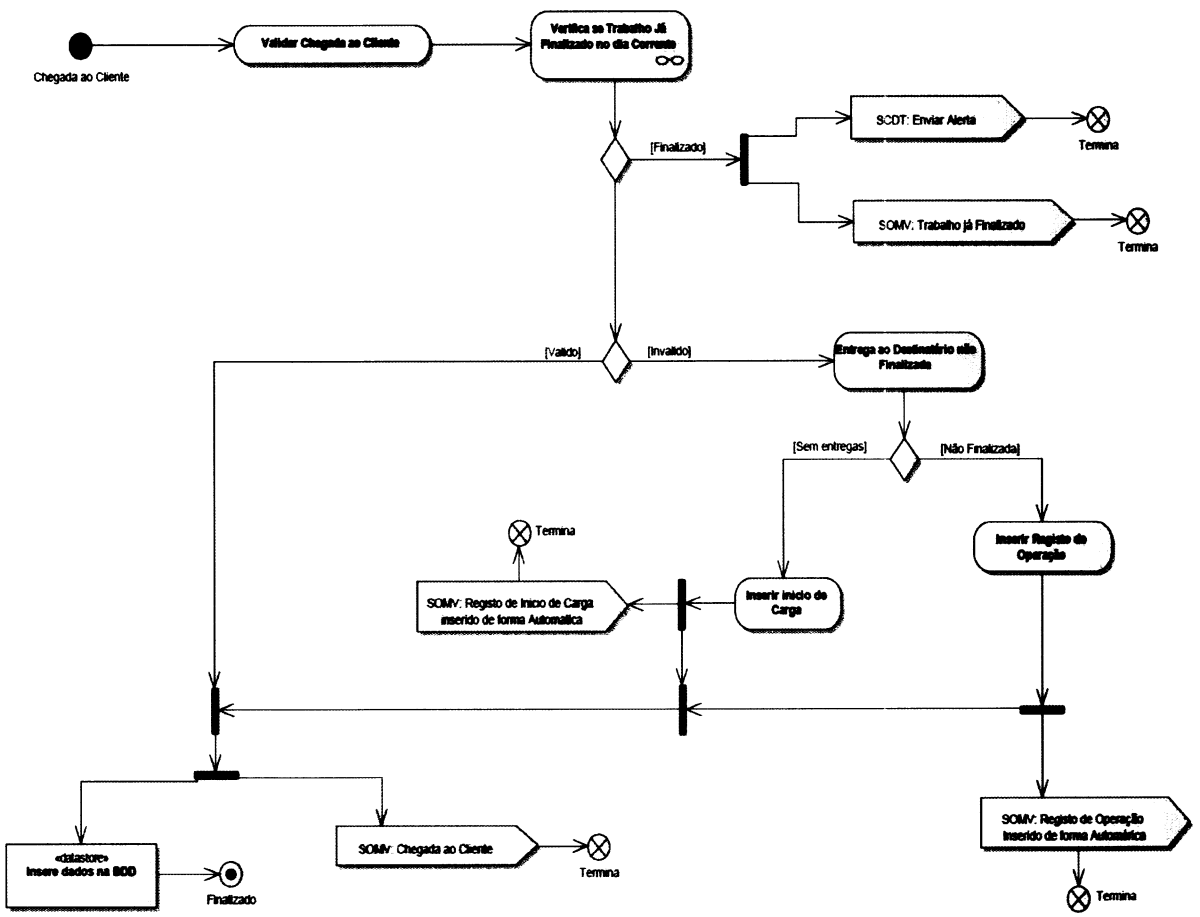
Anexo D – Diagrama de Actividade de "Inicio de Viagem"



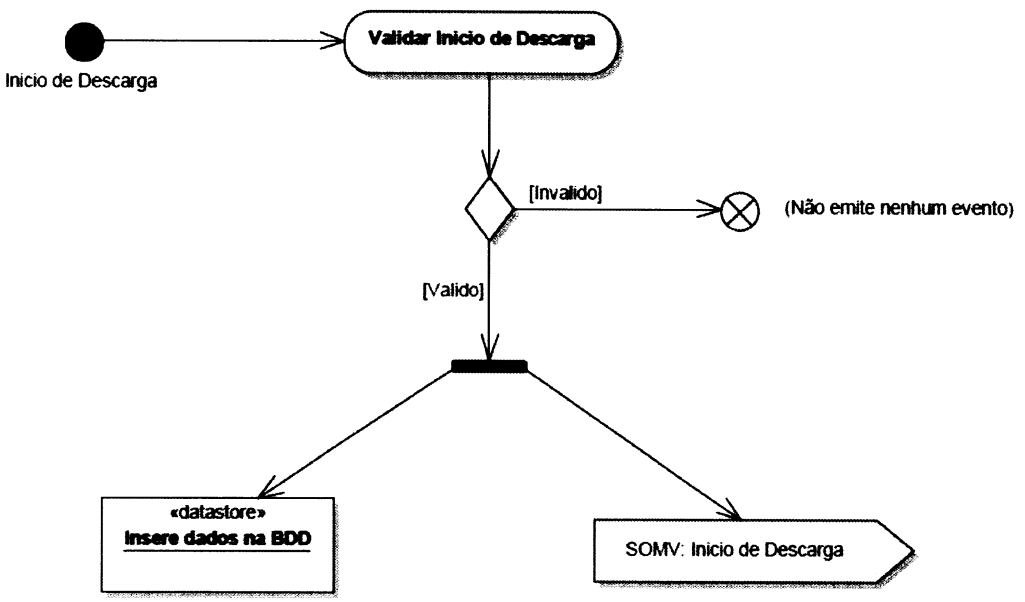
Anexo E – Diagrama de Actividade de "Inicio de Carga"



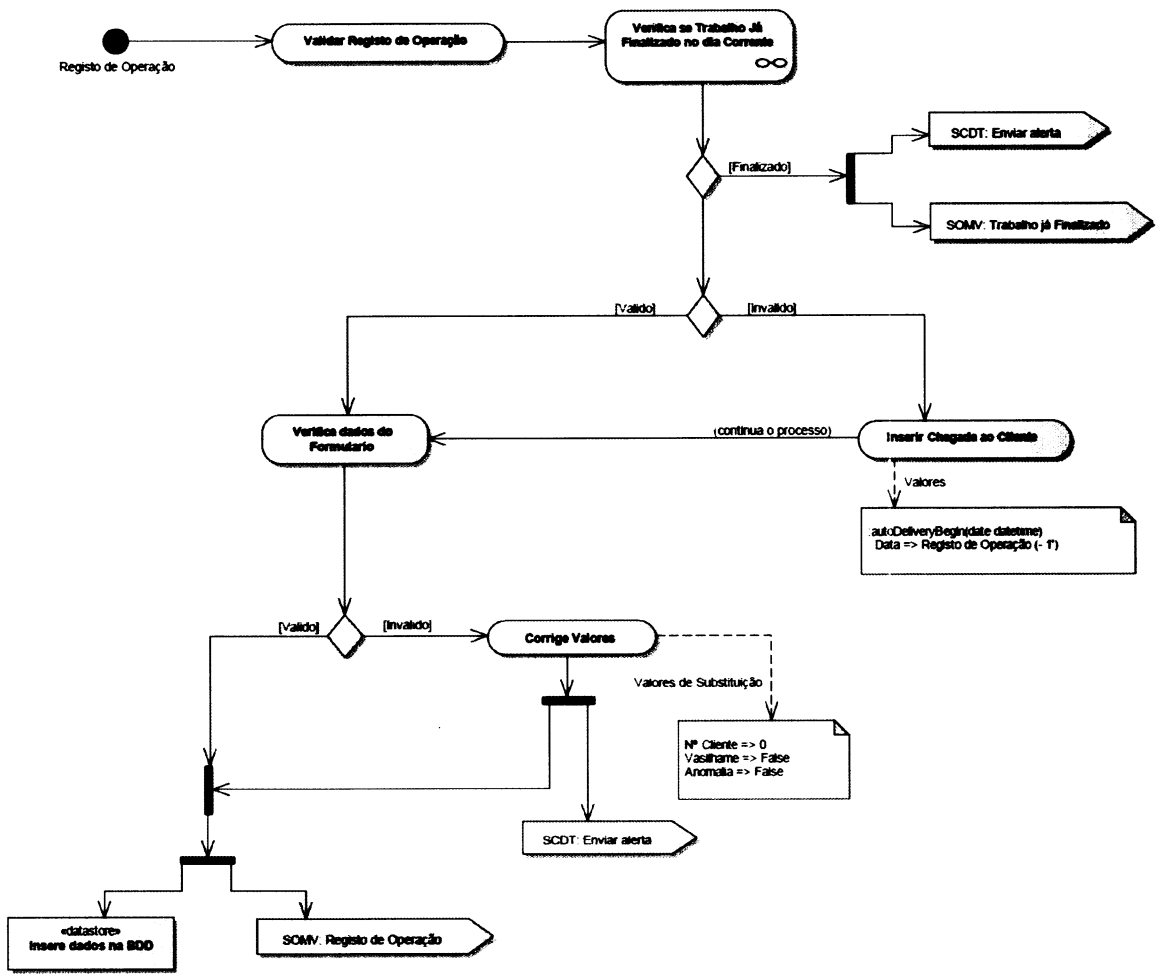
Anexo F – Diagrama de Actividade de "Chegada ao Cliente"



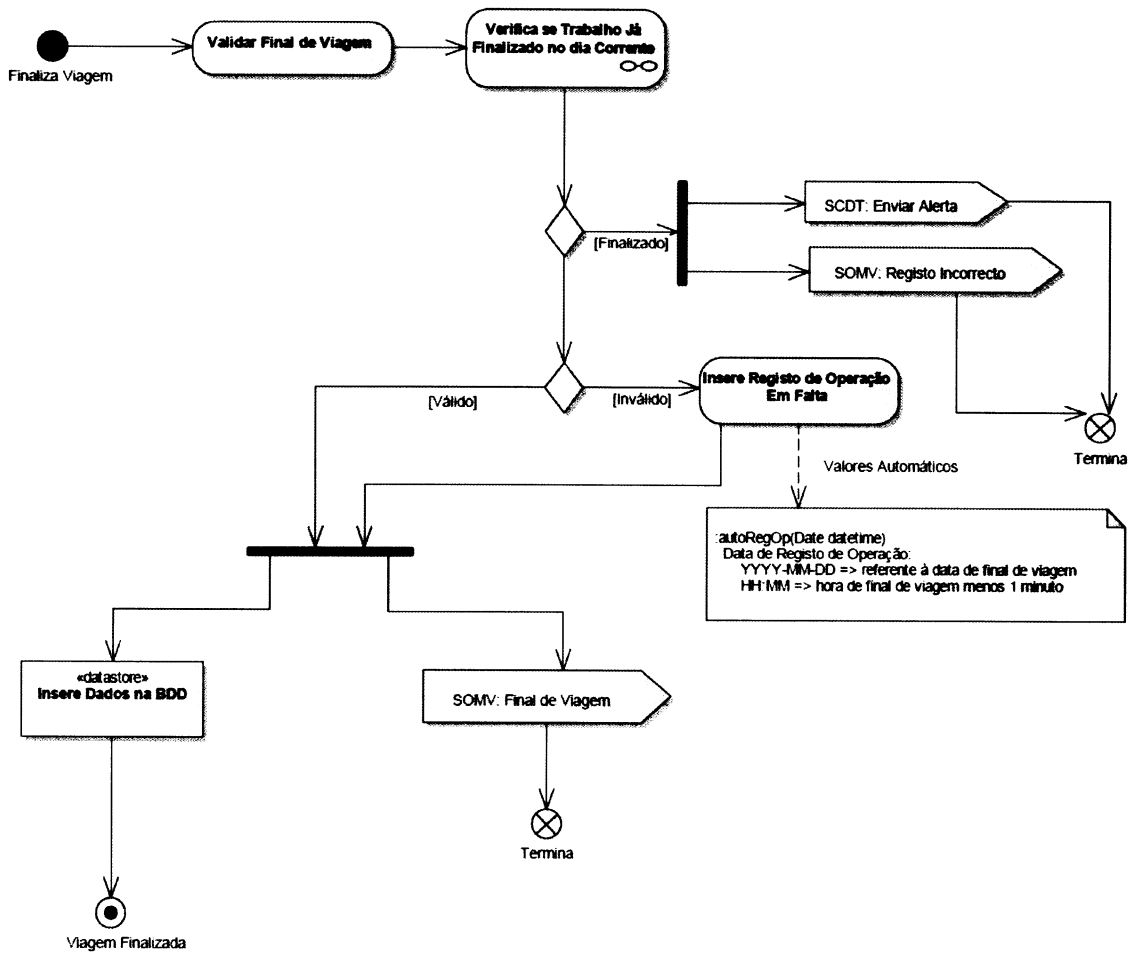
Anexo G – Diagrama de Actividade de "Inicio de Descarga"



Anexo H – Diagrama de Actividade de "Registo de Operação"



Anexo I – Diagrama de Actividade de "Fim de Viagem"



Anexo J – Diagrama de Actividade de "Fim de Trabalho"