

**DANIEL KONISHI
FABIO LUIZ PERRELLA
GILSON ALVES DA COSTA**

SISTEMA DE CONTROLE DE ENTREGAS

Monografia apresentada à
Escola Politécnica da
Universidade de São Paulo para
obtenção do Diploma de
Graduação em Engenharia

**São Paulo
2004**

**DANIEL KONISHI
FABIO LUIZ PERRELLA
GILSON ALVES DA COSTA**

SISTEMA DE CONTROLE DE ENTREGAS

Monografia apresentada à
Escola Politécnica da
Universidade de São Paulo para
obtenção do Diploma de
Graduação em Engenharia

Área de Concentração:
Engenharia de Computação

Orientador: Prof Dr. Paulo
Sérgio Muniz Silva

**São Paulo
2004**

FICHA CATALOGRÁFICA

Perrella, Fabio Luiz

**Sistema de controle de entregas / F.L. Perrella, D. Konishi,
G.A. da Costa. -- São Paulo, 2004.**

p.

**Trabalho de Formatura - Escola Politécnica da Universidade
de São Paulo. Departamento de Engenharia de Computação e
Sistemas Digitais.**

**1.Controle (Teoria de sistema e controle) I.Costa, Gilson Al-
ves da II.Konishi, Daniel III. Universidade de São Paulo. Escola
Politécnica. Departamento de Engenharia de Computação e
Sistemas Digitais IV.t.**

AGRADECIMENTOS

Seria impossível imaginar a realização de um trabalho de formatura apenas com o esforço e dedicação dos membros que compõe o grupo. Durante a realização desse trabalho contamos a ajuda e apoio do Professor Doutor Paulo Sérgio Muniz Silva ao qual deixamos os nossos sinceros agradecimentos.

Além disso, agradecemos a toda dedicação que tivemos dos professores do departamento de engenharia da computação e sistemas digitais durante esses cinco anos de curso.

RESUMO

O presente trabalho reúne informações do projeto de um Sistema de Controle de Entregas. Esse projeto foi desenvolvido segundo o Modelo Cascata de Desenvolvimento de Software aplicando padrões de documentos utilizados no laboratório de Engenharia de Software do curso de Engenharia de Computação da EPUSP. O sistema tem como principal objetivo auxiliar o processo de entregas de uma empresa. Com ele, a empresa pode ter um melhor controle sobre sua frota, acarretando uma melhora do serviço para seus clientes. A preocupação principal do corpo do trabalho está centrada descrever em um nível mais alto de abstração as características do sistema. Os detalhes estão descritos em documentos específicos, em anexos. O texto apresenta também assuntos conceituais. Além do cumprimento da formalidade de um documento de conclusão de curso, esse texto tem a pretensão de transmitir alguma experiência da prática de projeto, vivida pelos autores, a futuros projetistas de sistemas de software.

ABSTRACT

This work gather information about the development of a Delivery Control System. This project was developed following the standard waterfall model for system developement. Ours artefacts was based in the pattern used in the Laboratory in Software Engineering of the undergraduate course of Computer Engineering of EPUSP. The aim of the system is to improve the delivery process of a company.

With this system the company could improve the control of its vehicles, offering a better service for their customers.

The focus of this work is to outline the major property of the system. For further details you should look in the attached paper. In this paper we also put some basic ideas about delivery process. This document not only try to fullfill the requiment of a closing course paper but also try to show our own experience in system designing to the next system designers.

SUMÁRIO

LISTA DE FIGURAS.....	1
LISTA DE ABREVIATURAS E SIGLAS	2
1. INTRODUÇÃO.....	3
2. OBJETIVOS.....	4
3. JUSTIFICATIVA	5
3.1. MOTIVAÇÃO	5
3.2. APLICAÇÃO PRÁTICA.....	5
3.3. COMPARAÇÃO COM ALTERNATIVAS JÁ EXISTENTES.....	5
3.4. POTENCIAL DE EVOLUÇÃO.....	5
4. CONCEITOS	6
4.1. PROCESSO DE LOGÍSTICA.....	6
4.2. PDAS.....	6
4.3. REDES WI-FI.....	7
4.4. PLATAFORMA DOT NET	7
4.5. WEB SERVICES	8
4.6. DESIGN PATTERNS.....	8
5. PROCESSO DE TRANSPORTE DE MERCADORIAS.....	13
5.1. DESCRIÇÃO.....	13
5.2. ATIVIDADES - TRANSPORTE DE MERCADORIAS	14
5.3. ESCOPO DO SCE (WISH LIST).....	15
6. DETALHAMENTO.....	16
6.1. SOFTWARE.....	16
6.2. CASOS DE USO.....	17
6.3. ARQUITETURA DE SOFTWARE	19
6.3.1. MODELO DE COMPONENTES DISTRIBUÍDOS	19
6.3.1.1. CAMADA DE APRESENTAÇÃO	19

6.3.1.2.	CAMADA DE NEGÓCIOS	20
6.3.1.3.	CAMADA DE DADOS	20
6.3.1.4.	COMPONENTES DE ACESSO A DADOS	20
6.3.1.5.	COMPONENTES DA ENTIDADE DE NEGÓCIO	22
6.4.	CLASSES	24
6.5.	INTERFACE.....	26
6.6.	BANCO DE DADOS	26
6.7.	HARDWARE.....	27
6.7.1.	ARQUITETURA DO SCE.....	27
7.	PLANEJAMENTO E METODOLOGIA.....	29
7.1.	METODOLOGIA	29
7.2.	ETAPAS DO MODELO CASCATA	30
7.3.	CRONOGRAMA	31
7.4.	APLICAÇÃO DO MODELO CASCATA AO PROJETO	32
8.	CONCLUSÃO.....	33
9.	LISTA DE REFERÊNCIAS	34
	ANEXO A – ESPECIFICAÇÃO DE CASOS DE USO	35
1.	GERIR FUNCIONÁRIO	35
1.1.	LISTAR FUNCIONÁRIOS	35
1.2.	CONSULTAR FUNCIONÁRIO.....	35
1.3.	ATRIBUIR PERMISSÃO DE ACESSO	36
2.	GERIR AGENTES EXTERNOS	36
2.1.	INSERIR AGENTE EXTERNO	36
2.2.	LISTAR AGENTES EXTERNOS.....	37
2.3.	CONSULTAR AGENTE EXTERNO	37
2.4.	ALTERAR AGENTE EXTERNO.....	37
3.	GERIR VEÍCULOS	38
3.1.	INSERIR VEÍCULO.....	38

3.2.	LISTAR VEÍCULOS.....	39
3.3.	CONSULTAR VEÍCULO.....	39
3.4.	ALTERAR VEÍCULO	39
4.	GERIR TIPOS DE OCORRÊNCIAS.....	40
4.1.	INSERIR TIPO DE OCORRÊNCIA.....	40
4.2.	LISTAR TIPOS DE OCORRÊNCIA.....	41
4.3.	ALTERAR TIPO DE OCORRÊNCIA	41
5.	GERIR TIPOS DE ORIENTAÇÕES	42
5.1.	INSERIR TIPO DE ORIENTAÇÃO.....	42
5.2.	LISTAR TIPOS DE ORIENTAÇÃO	42
5.3.	ALTERAR TIPO DE ORIENTAÇÃO	42
6.	GERIR ENTREGAS	43
6.1.	LISTAR ENTREGAS	43
6.2.	CONSULTAR ENTREGA.....	43
6.3.	CONSULTAR PEDIDO	44
6.4.	ENVIAR ORIENTAÇÃO	44
7.	VERIFICAR NOVOS PEDIDOS.....	44
8.	VERIFICAR CHEGADA DE NOVAS OCORRÊNCIAS.....	45
9.	VISUALIZAR ENTREGAS	45
9.1.	LISTAR ENTREGAS	45
9.2.	CONSULTAR ENTREGAS	45
9.3.	CONSULTAR PEDIDO	46
9.4.	ENVIAR OCORRÊNCIAS	46
	ANEXO B – MODELO DE CLASSES.....	47
10.	ENTITIES.....	47
10.1.	VEHICLEINFO.....	47
10.2.	EMPLOYEEINFO	47
10.3.	EXTERNALAGENTINFO	47

10.4.	ORIENTATIONTYPEINFO	48
10.5.	OCCURTYPEINFO.....	48
10.6.	ORIENTATIONINFO.....	48
10.7.	OCCURINFO	48
10.8.	DSDELIVERYLISTINFO	48
10.9.	DELIVERYINFO	49
10.10.	DSORDERLISTINFO	49
10.11.	DSOCCURLISTINFO	49
11.	BUSINESS ROLE.....	50
11.1.	EMPLOYEEBR.....	50
11.2.	EXTERNALAGENTBR.....	50
11.3.	VEHICLEBR	51
11.4.	ORIENTATIONTYPEBR.....	52
11.5.	OCCURBR.....	53
11.6.	DELIVERYBR.....	53
	ANEXO C – MODELO DINÂMICO	56
	ANEXO D – PROJETO DA INTERFACE DE USUÁRIO	59
1.	CARACTERIZAÇÃO DO USUÁRIO	59
2.	ESTILO DA INTERFACE	59
3.	PROTÓTIPO DE NAVEGAÇÃO.....	60
3.1.	DIAGRAMA HIERÁRQUICO DAS PÁGINAS	60
	ANEXO E – PROJETO DE BANCO DE DADOS.....	62
1.	TABELAS DO MODELO RELACIONAL	62
	ANEXO F – PLANO DE ACEITAÇÃO	63
1.	OBJETIVO DO DOCUMENTO.....	63
2.	RESUMO DO TESTE DE ACEITAÇÃO	63

3. INFRA-ESTRUTURA PARA A ACEITAÇÃO	64
4. LISTA DE TESTES.....	64
5. TESTES	65
ANEXO G – PADRÃO DE CODIFICAÇÃO	74
1. DEFINIÇÃO DE FUNÇÕES.....	74
2. CONVENÇÃO DE NOMES.....	74

LISTA DE FIGURAS

FIGURA 1 – DIAGRAMA DE ATIVIDADES DO PROCESSO DE TRANSPORTE DE MERCADORIAS.....	13
FIGURA 2 – DIAGRAMA DE PACOTES	16
FIGURA 3 - DIAGRAMA DE CASOS DE USO.....	18
FIGURA 4 - CAMADAS DE UMA APLICAÇÃO DISTRIBUÍDA	19
FIGURA 5 – DIAGRAMA DE CLASSES	24
FIGURA 6 - DIAGRAMA DE CLASSES (NEGÓCIO).....	25
FIGURA 7 - CLASSES DE ENTIDADE DE NEGÓCIO	26
FIGURA 8 - ARQUITETURA DO SISTEMA.....	27
FIGURA 9 – MODELO CASCATA DE PROJETO DE SOFTWARE.....	29
FIGURA 6 – ESQUEMA HIERÁRQUICO DAS PÁGINAS	60
FIGURA 7 – LAYOUT DA INTERFACE	61

LISTA DE ABREVIATURAS E SIGLAS

LAN	- Local Área Network
PDA	- Personal Digital Assistat
RUP	- Rational Unified Process
SCE	- Sistema de Controle de Entregas
SOAP	- Simple Object Access Protocol
UDDI	- Universal Description, Discovery and Integration
XML	- Extensible Markup Language
W3C	- World Wide Web Consortium
WI-FI	- Wireless Fidelity
WSDL	- Web Services Description Language

1. INTRODUÇÃO

Segundo artigo publicado na ComexNet, a logística está muito em voga ultimamente pois é considerada a última fronteira, ainda não explorada, para a redução de custos de qualquer produto. Existem várias definições para logística e uma das mais resumidas diz que a logística é a rede de facilidades montada para movimentar materiais e/ou produtos acabados. Outra definição interessante e mais abrangente diz o seguinte: trata-se do processo de planejamento, implementação e controle do fluxo e armazenagem de matérias prima, inventário em processo, produtos acabados e informações correlatas do ponto de origem ao ponto de consumo em conformidade com os requisitos do cliente.

Investir em logística é uma estratégia na qual empresas de transportes de carga no Brasil estão apostando pra enfrentar a virada do milênio. Em outras partes do mundo como EUA, Europa e alguns países da Ásia, a logística já se encontra num processo bastante avançado.

2. OBJETIVOS

- Ter um contato mais direto com a área de modelagem e análise de sistemas de software;
- Utilizar uma tecnologia móvel, aprendendo seu modo de utilização e como implementá-la;
- Implementar um sistema de software utilizando a tecnologia .NET e ferramentas de desenvolvimento como Visio, Project, Source Safe da Microsoft.

Assim, pretende-se alcançar esses objetivos através de um estudo de caso: um processo de transporte de mercadorias. Esse processo é parte de um processo mais abrangente denominado Logística.

3. JUSTIFICATIVA

3.1. *Motivação*

A principal motivação do projeto é poder participar de um ciclo completo de elaboração de uma solução para um processo real, nesse caso, o processo de transporte de mercadorias, pois as questões e dúvidas aqui levantadas dificilmente serão discutidas adequadamente em um ambiente empresarial.

3.2. *Aplicação Prática*

O resultado do projeto de formatura será um Sistema de Controle de Entregas (SCE), que poderá ser utilizado por uma empresa que possua uma frota de veículos que realizam entregas.

Com o sistema implementado, essa empresa poderá monitorar o estado de sua frota, obter previsões de entregas de seus pedidos, disponibilizar a informação aos clientes, observar a produtividade de seus funcionários e tomar decisões sobre sua frota, enviando orientações e recebendo ocorrências dos condutores dos veículos.

3.3. *Comparação com alternativas já existentes*

A maioria dos sistemas de controle de entregas já existentes no mercado utiliza a tecnologia GPS para o monitoramento da frota, oferecendo informações de localização precisas e em tempo real. No entanto, seu custo é muito elevado e a tecnologia é pouco acessível. Além do mais, em muitos casos, não é de grande importância saber a localização exata do veículo, bastando apenas saber o estado atual e uma previsão dos horários de entrega dos pedidos. Por isso, foi escolhida para a implementação do projeto uma tecnologia móvel mais acessível, tanto em custo, como em viabilidade para desenvolvimento: Pocket PC conectado à Internet através de rede Wi-Fi.

3.4. *Potencial de Evolução*

Uma versão futura e mais elaborada do sistema poderia controlar a central de atendimento das ocorrências, ter um módulo de controle de expedição e de estoque, definir políticas de logística, roteirização, entre outros recursos.

4. CONCEITOS

4.1. *Processo de Logística*

Segue-se uma descrição resumida das atividades de um processo de Logística.

“Antes de embarcar uma mercadoria, qualquer empresa precisa processar informações, desenvolver fornecedores, acionar compras, receber e verificar materiais, embalar e movimentar produtos, estocando-os apropriadamente para preservar sua integridade. É preciso ainda, planejar e controlar estoques e produção, movimentar e estocar mercadorias, otimizar *layouts* e fluxos de materiais e pessoas, qualificar colaboradores e parceiros, medir e gerenciar custos, avaliar e auditar a qualidade, entre outros”.(GASNIER, D., 2004)

4.2. *PDA's*

Segundo THE WIRELESS LAN BIBLE, Um PDA (*Personal Digital Assistant*), ou Assistente Pessoal Digital, é um computador de dimensões reduzidas, dotado de grande capacidade computacional, cumprindo as funções de agenda e sistema de informação de escritório, com possibilidade de interconexão com um computador pessoal e uma rede informática sem fios - Wi-Fi - para acesso a correio eletrônico e internet.

Os PDA's de hoje possuem grande quantidade de memória e diversos softwares para várias áreas de interesse.

Os modelos mais sofisticados possuem modem (para acesso à Internet), câmera digital acoplada, tela colorida e rede sem fio embutida.

Os PDA's trazem das agendas eletrônicas somente as dimensões, pois sua utilidade e aplicabilidade estão se aproximando cada vez mais rapidamente dos computadores de mesa. Existem duas famílias principais de PDA no mercado hoje: O Palm e o Pocket PC. Os Palm utilizam o sistema operacional Palm OS da PalmSource (ex-Palm Computing), um sistema rápido e bastante confiável, que, mesmo restrito quanto ao número de fabricantes que o adotaram, é atualmente o mais utilizado no mundo.

Os Pocket PC utilizam o sistema Pocket PC (antigo Windows CE) da Microsoft, compatível com o Windows e adotado por uma gama bem variada de fabricantes de

PDA. Estes PDAs utilizam programas que possibilitam a troca de dados com programas Windows dos computadores de mesa.

O número de PDAs no mundo vem crescendo de forma exponencial, mas tendências indicam que em poucos anos os SmartPhones (desenvolvido através da "fusão" entre um PDA e um telefone celular) serão maioria absoluta.

4.3. Redes Wi-Fi

Segundo THE WIRELESS LAN BIBLE, o Wi-Fi, abreviação de "Wireless Fidelity", é um conjunto de padrões para redes locais sem fio baseados na especificação IEEE 802.11.

O Wi-Fi foi desenvolvido para ser usado em redes LAN, mas atualmente está sendo usado também para acesso à Internet. Esse tipo de rede permite uma pessoa com um computador, ou um PDA, com dispositivo de conexão sem fio, conectar-se à Internet quando próximo de um *Acces Point* (Ponto de Acesso), que são pontos de transmissão do sinal de rádio.

4.4. Plataforma Dot Net

Segundo a Microsoft, o Dot Net é uma plataforma de software que conecta informações, sistemas, pessoas e dispositivos. A plataforma Dot Net conecta uma grande variedade de tecnologias de uso pessoal e de negócios, de telefones celulares a servidores corporativos, permitindo o acesso a informações importantes, onde e sempre que forem necessárias.

Desenvolvido sobre os padrões de *Web Services XML*, o Dot Net possibilita que sistemas e aplicativos, novos ou já existentes, conectem seus dados e transações, independentemente do sistema operacional, tipo de computador ou de dispositivo móvel que sejam utilizados, ou mesmo da linguagem de programação utilizada na sua criação.

O Dot Net está presente em toda a linha de produtos Microsoft, oferecendo a capacidade de desenvolver, implementar, gerenciar e usar soluções conectadas através de *Web Services XML*, de maneira rápida, barata e segura. Essas soluções permitem uma integração mais rápida e ágil entre os negócios e o acesso a informações a qualquer hora, em qualquer lugar e em qualquer dispositivo.

4.5. Web Services

Segundo a Microsoft, os *Web Services* são uma tecnologia para negócios na Web. Os *Web Services* XML usam tecnologias programáveis e reutilizáveis que aproveitam a flexibilidade da Internet. Com eles é possível ter uma infinidade de aplicativos conectados em rede, mesmo rodando em plataformas diferentes, fornecendo informações a todos os seus clientes, parceiros e funcionários.

Os *Web Services* baseiam-se em um conjunto de padrões abertos, incluindo XML, SOAP, WSDL e UDDI, controlados pelo W3C.

4.6. Design Patterns

Introdução

Os projetistas de sistemas devem encontrar os objetos e fatorá-los em classes no nível correto de granularidade, definindo as interfaces das classes e as hierarquias de herança, estabelecendo as relações chaves entre eles, criando projetos específicos para resolver o problema, mas também genérico o suficiente para atender futuros problemas e requisitos (GAMMA, 2000). Os objetos bem modelados fazem com que os programadores ganhem mais tempo na implementação. Depois de modeladas as classes bases e testadas, os desenvolvedores precisam apenas se preocupar com a implementação da especialização da subclasse criada (ALUR, 2002).

Para se poder reutilizar uma maior quantidade de soluções, não basta ter escrito classes que descrevem toda a complexidade do projeto. Os projetistas novatos tendem a usar técnicas não orientadas a objetos pela sobrecarga de opções disponíveis. Sistemas cada vez mais complexos tendem a ter um número de classes cada vez maior, dificultando o desenvolvedor na codificação.

Com o passar dos anos de experiência dos arquitetos em projetar soluções, observou-se que a essência dos problemas é semelhante e que quando especializavam as classes bases, o que é difícil conseguir na primeira vez, ganhavam maior

produtividade no desenvolvimento, fazendo a reutilização de soluções corretas e que comprovadamente funcionavam (GAMMA, 2000) (SUN MYCROSYSTEM, 2002).

Observando que na essência os problemas assemelham-se, os projetistas começaram a documentar as soluções criadas durante os anos. Essa documentação foi sendo aprimorada formando padrões (*patterns*) de desenvolvimento. Os padrões, segundo Alur (2002), permitem documentar um problema conhecido e recorrente (que surgem várias vezes durante o projeto) e sua solução em um contexto específico e comunicar esse conhecimento para outras pessoas. Cada padrão tem uma aplicação específica dentro da arquitetura do software composto de multicamadas que separam a lógica de negócio, da interface do usuário e do acesso aos dados persistentes.

Patterns

Na área da informática, a evolução das capacidades computacionais e os seus requisitos crescem em ritmo acelerado. Dado que a capacidade de processamento duplica a cada 18 meses, segundo a lei de Moore, Pressman (1995) pergunta como a Engenharia de Software é capaz de responder a este crescente desenvolvimento

A primeira solução aparece com a reutilização de código e de soluções anteriormente implementadas, com a intenção de aumentar a rapidez com que são criadas novas aplicações e de aprimorar sua qualidade final. Mas mesmo assim, não é evitado um problema: “Quem desenvolve software tem muitas vezes a sensação de que está reinventando a roda“. Dentro da mesma empresa pode estar um desenvolvedor tentando resolver um problema, enquanto que o colega no cubículo ao lado já tem a solução há meses.

A solução encontrada originou-se da adaptação dos conceitos de padrões de arquitetura, desenvolvida por Christopher Alexander, à área da Engenharia de Software. Os padrões aparecem como um método para facilitar essa comunicação de conhecimentos entre entidades e indivíduos. Cada padrão descreve um problema que ocorre repetidas vezes em nosso ambiente, e então descreve o núcleo da solução para aquele problema, de tal maneira que se pode usar essa solução milhões de vezes sem

nunca fazê-la da mesma forma duas vezes segundo Christopher Alexander (apud GAMMA, 2000).

Os padrões abrangem diferentes níveis de abstração, resultantes da enorme aplicabilidade prática que possuem, podendo classificá-los em diversas categorias, de modo a facilitar a sua recuperação e utilização num determinado problema. No entanto, esta classificação é um pouco flexível, podendo haver padrões que nitidamente se encaixam em várias categorias. Na tabela 1 são descritas as categorias mais importantes de padrões, segundo Alur (2002).

Padrão	Explicação
Padrões de processo	Definem soluções para os problemas encontrados nos processos envolvidos na engenharia de Software: desenvolvimento, controle de configuração, testes, etc.
Padrões arquiteturais	Exprimem a organização das estruturas fundamentais de sistemas de software ou hardware.
Padrões de padrão (do Inglês, <i>patterns on patterns</i>)	São padrões que descrevem como um padrão deve ser escrito, ou seja, padrões que uniformizam o modo como estes devem ser apresentados aos seus utilizadores. Padrões de análise descrevem soluções para os problemas de análise de sistemas, incluindo conhecimento sobre o domínio específico de aplicação.
Padrões de projeto	Definem soluções para diversos problemas de projeto de software. Padrões de interface definem soluções para problemas comuns no projeto de interfaces de sistemas. É um caso particular dos padrões de projeto.
Padrões de programação	Descrevem soluções particulares de programação numa determinada linguagem ou regras gerais de estilo de programação.
Padrões de persistência	Descrevem soluções para problemas de armazenamento de informações em arquivos ou bancos de dados.
Padrões de Hipertexto	Descrevem soluções para problemas que se encontram no projeto de aplicação direcionadas para a World Wide Web (WWW).

Para facilitar a utilização e divulgação dos padrões de projeto, existe uma catalogação que difere entre alguns autores sob o aspecto de propósito e intenção de utilização. Um catálogo de padrões pode oferecer um esquema de classificação e recuperação dos seus padrões, já que eles estão subdivididos em categorias, e adiciona uma certa quantidade de estrutura e organização a uma coleção de padrões, mas tipicamente não vai além de mostrar apenas as estruturas e relações mais superficiais (quando o faz). Na figura 1 tem-se a classificação feita por Gamma (2000), classificado pelo seu escopo e um determinado propósito.

Para facilitar a utilização e divulgação dos padrões de projeto, existe uma catalogação que difere entre alguns autores sob o aspecto de propósito e intenção de

utilização. Um catálogo de padrões pode oferecer um esquema de classificação e recuperação dos seus padrões, já que eles estão subdivididos em categorias, e adiciona uma certa quantidade de estrutura e organização a uma coleção de padrões, mas tipicamente não vai além de mostrar apenas as estruturas e relações mais superficiais (quando o faz). Na figura 1 tem-se a classificação feita por Gamma (2000), classificado pelo seu escopo e um determinado propósito.

		Propósito		
		1. Criação	2. Estrutura	3. Comportamento
Escopo	Classe	<i>Factory Method</i>	<i>Class Adapter</i>	<i>Interpreter</i> <i>Template Method</i>
	Objeto	<i>Abstract Factory</i> <i>Builder</i> <i>Prototype</i> <i>Singleton</i>	<i>Object Adapter</i> <i>Bridge</i> <i>Composite</i> <i>Decorator</i> <i>Facade</i> <i>Flyweight</i> <i>Proxy</i>	<i>Chain of Responsibility</i> <i>Command</i> <i>Iterator</i> <i>Mediator</i> <i>Memento</i> <i>Observer</i> <i>State</i> <i>Strategy</i> <i>Visitor</i>

Aplicação de Patterns no desenvolvimento de software

Segundo Buschmann (1998), os padrões não definem um novo método para o desenvolvimento de software que substitua os já existentes. Eles apenas complementam os métodos de análise e projeto gerais e independentes do problema, por exemplo, Booch, OMT, Shlaer/Mellor, etc, com diretrizes para resolver problemas específicos e concretos. Buschmann (1998) sugeriu os seguintes passos para desenvolver um sistema usando padrões de software:

- a) utilizar o método preferido para o processo de desenvolvimento de software em cada fase do desenvolvimento;
- b) utilizar um sistema de padrões adequado para guiar o projeto e a implementação de soluções para problemas específicos, isto é, sempre que se encontrar um padrão que resolva um problema do projeto presente no sistema, utilizar os passos da implementação associados a esse padrão. Se esses se referirem aos outros padrões, aplicam-se recursivamente;

c) se o sistema de padrões não incluir um padrão para o problema do projeto, tenta-se

encontrar um padrão em outras fontes conhecidas;

d) se nenhum padrão estiver disponível, aplica-se as diretrizes de análise e projeto do

método que se está usando. Essas diretrizes fornecem pelo menos algum apoio útil para resolver o problema do projeto em questão. Segundo Buschmann (1998), essa abordagem simples evita que se criem outros métodos de projeto. Ela combina a experiência de desenvolvimento de software captada pelos métodos de análise e projeto com as soluções específicas para problemas de projeto descritas pelos padrões.

5. PROCESSO DE TRANSPORTE DE MERCADORIAS

5.1. Descrição

O diagrama de atividades que segue, ilustra de modo simplificado, o processo de transporte de mercadorias - parte integrante do processo de Logística - que é o processo a ser automatizado pelo SCE.

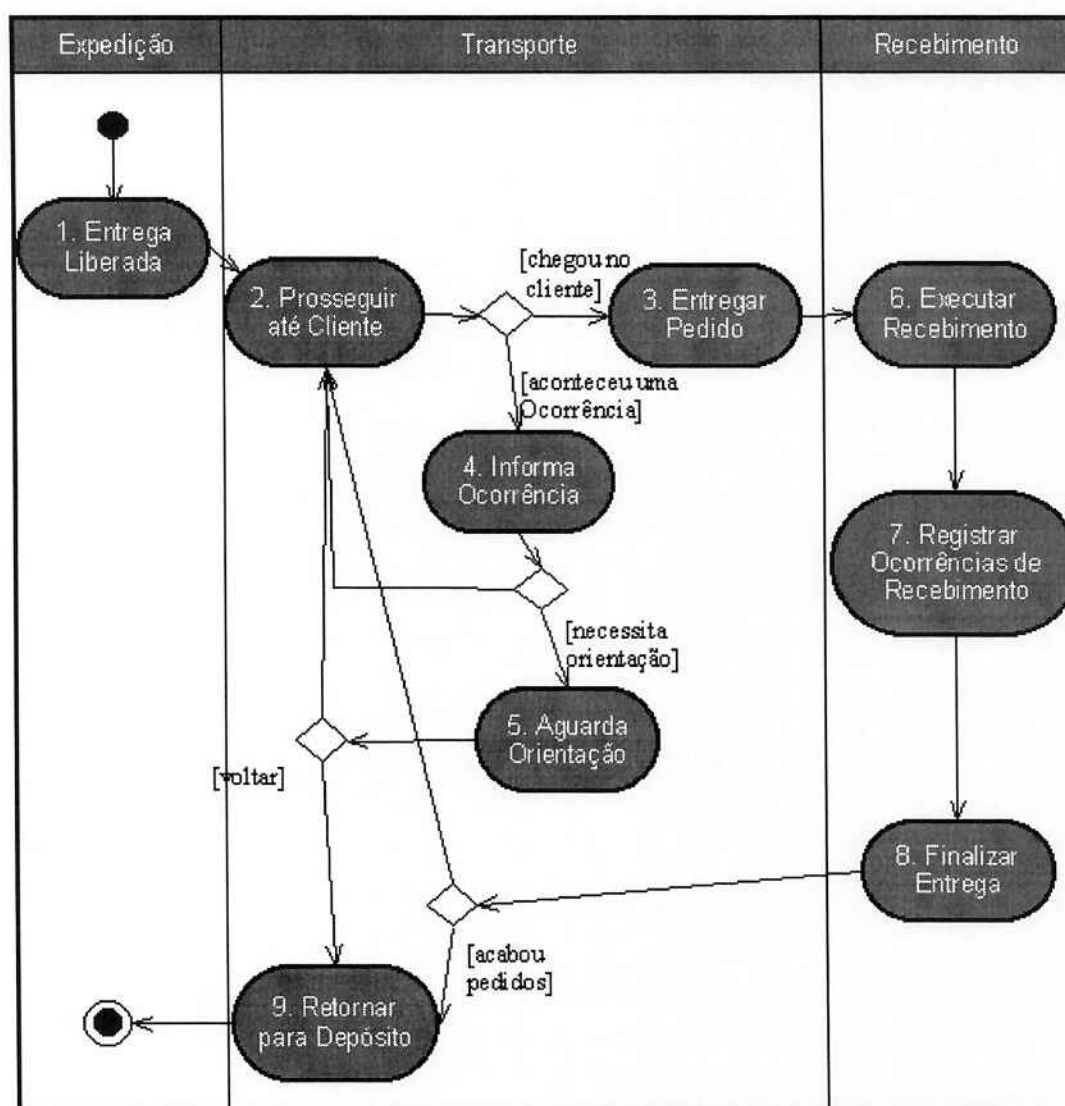


Figura 1 – Diagrama de Atividades do processo de Transporte de Mercadorias

Descrição de alguns termos utilizados na figura 1:

- **Ocorrência:** é algum fato que acontece durante o processo de transporte de mercadorias, que pode ser usado para justificar resultados finais da entrega ou indicar seu estado atual.
- **Orientação:** são ações geradas pelos responsáveis da área de logística da empresa, com o objetivo de resolver alguma situação provinda de uma ocorrência.

5.2. Atividades - Transporte de Mercadorias

Seguem as descrições das atividades da Figura 1. Os índices são os mesmos utilizados na figura 1.

	Atividade	Descrição
1	Liberar Nova Entrega	Momento que um veículo se encontra pronto para efetuar a entrega, liberado pelo setor de expedição.
2	Prosseguir até o Cliente	Momento em que o veículo está se dirigindo ao cliente
3	Entregar Pedido	Momento em que o veículo chega ao cliente e efetua o procedimento de entrega
4	Informa Ocorrência	Quando acontece uma ocorrência, o motorista do veículo deve informar a alguém da empresa o que aconteceu.
5	Aguarda Orientação	Momento em que o motorista fica aguardando alguma orientação da empresa para saber o que deve ser feito em consequência de alguma ocorrência.
6	Executar Procedimento de Recebimento	Momento em que o funcionário responsável do cliente efetua seus procedimentos para receber uma entrega
7	Registrar Ocorrências de recebimento	Momento em que o funcionário do cliente registra as ocorrências relativas ao pedido que foi entregue, validando-o ou não.
8	Finalizar Entrega	Momento em que o motorista considera a entrega do pedido como finalizada
9	Retornar	Momento em que o veículo se dirige ao depósito

5.3. Escopo do SCE (Wish List)

Baseado nas atividades do processo de transporte de mercadorias, seguem as descrições das funcionalidades desejadas no SCE, com objetivo de automatizar essas atividades descritas na seção 5.2. Os índices referem-se aos mesmos utilizados na seção 5.2.

	Atividade	Descrição
1	Liberar Nova Entrega	O SCE deve receber informações do pedido do sistema da expedição. O SCE deve armazenar as informações relativas aos pedidos e à entrega que são relevantes.
3	Entregar Pedido	O SCE deve registrar o momento da entrega
4	Informa Ocorrência	O SCE deve prover um meio estruturado para que sejam alimentadas ocorrências relativas às entregas. Estas entradas podem ser efetuadas pelos motoristas, funcionários da empresa ou funcionários do cliente. As ocorrências devem ser visíveis ao responsável pela parte de entregas da empresa, para que ele possa tomar decisões. Também devem ser visíveis para os clientes.
5	Aguarda Orientação	O SCE deve prover um meio estruturado para que as orientações sejam enviadas pelo responsável da área de transportes da empresa e os motoristas recebam-nas.
6	Executar procedimento de Recebimento	O SCE deve prover um meio para um funcionário do cliente enviar ocorrências.
7	Registrar Ocorrências de recebimento	O SCE deve prover um meio estruturado para o funcionário do cliente entrar as ocorrências relativas ao recebimento do pedido.
8	Finalizar Entrega	O SCE deve prover um meio estruturado para que seja entrada a informação de entrega finalizada.

6. DETALHAMENTO

Nesta seção, detalha-se o projeto, mostrando-se detalhes do projeto de software e as características do hardware.

6.1. Software

A seguir, serão exibidas informações relativas ao projeto de software do SCE.

Primeiramente, exibe-se o relacionamento do SCE com alguns outros sistemas que compõem o chamado sistema de *Back End*.

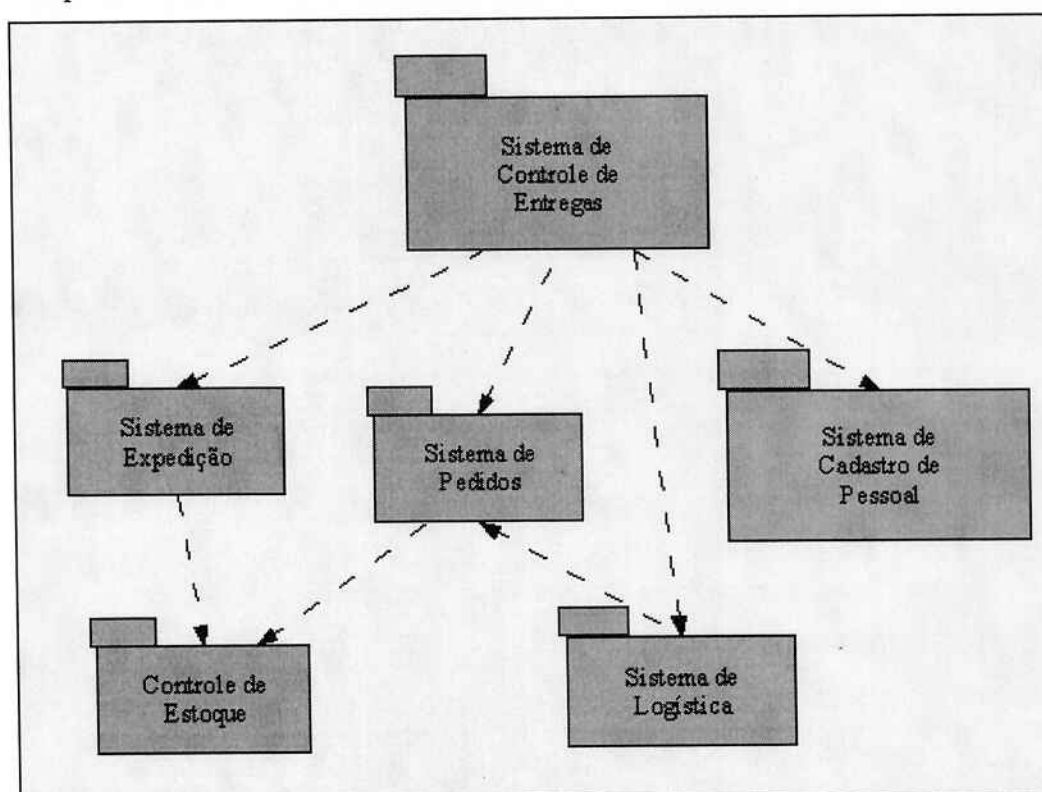


Figura 2 – Diagrama de Pacotes

Através da figura 2, pode-se perceber melhor quais as funcionalidades que não são do escopo do SCE, como cadastro de funcionários, de pedidos, de rota de entrega, etc.

Nesse projeto, a solução adotada para que se pudesse demonstrar todas as funcionalidades do SCE, foi simular algumas bases de dados dos sistemas de *back end*, como por exemplo, uma tabela com informações cadastrais de funcionários e uma tabela com os pedidos liberados pelo sistema de expedição.

6.2. Casos de Uso

Segue uma descrição sobre os atores do sistema:

- Gerente de Transporte: pessoa responsável pela área de transportes da empresa. É quem toma decisões sobre a frota.
- Auxiliar de Transporte: pessoa que auxilia gerente nas atividades.
- Motorista: motorista de um veículo
- Agente Externo: funcionário do cliente com acesso ao sistema
- Cron: programa que ativa algumas rotinas de tempos em tempos

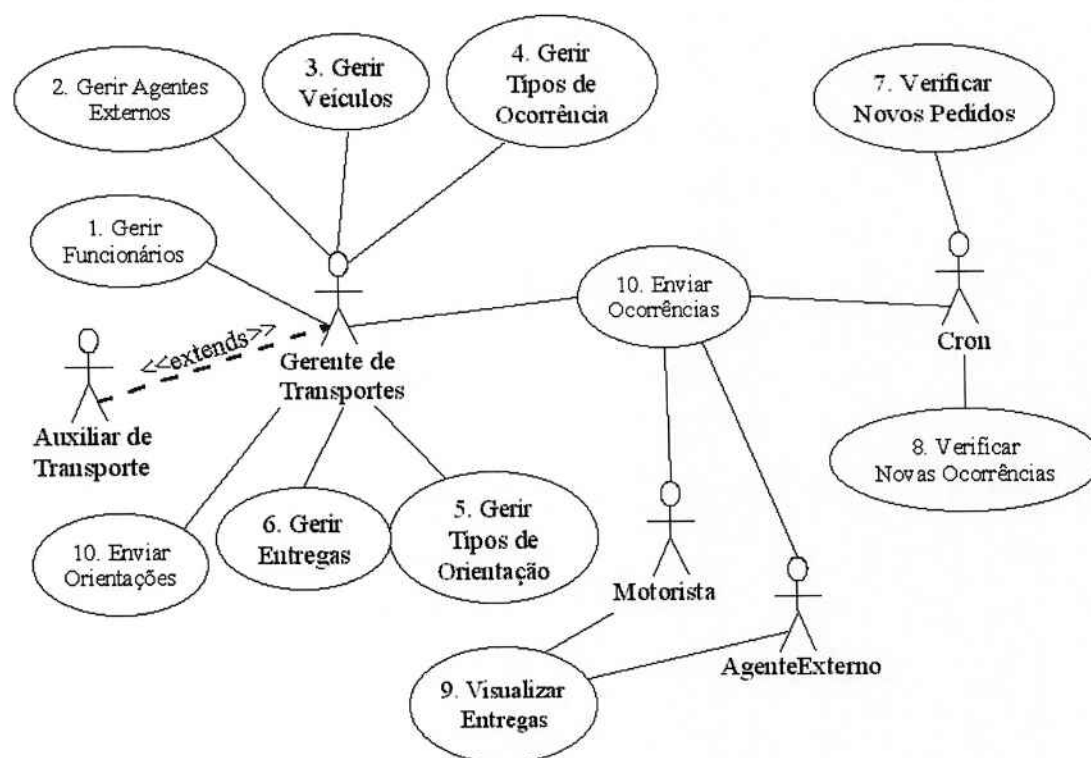


Figura 3 - Diagrama de Casos de Uso

Na figura 3, pode-se visualizar os papéis dos atores envolvidos no SCE. Os casos de uso denominados “Gerir x” se referem, no geral, as ações de cadastro, alteração, remoção, pesquisa e listagem de algum tipo de informação, com algumas exceções, por exemplo o caso de uso Gerir Funcionários, que não possui a ação de cadastro de funcionários, pelo motivo de que a inserção de informações cadastrais dos funcionários são de funcionalidades dos sistemas de *back end*.

Para maiores detalhes, uma descrição dos casos de uso encontra-se em anexo [A].

6.3. Arquitetura de Software

Essa seção descreve o modelo da arquitetura de software utilizada no sistema.

6.3.1. Modelo de Componentes Distribuídos

O projeto é baseado em um padrão de camadas sugerido pela Microsoft. Esse padrão sugere a aplicação de componentes distribuídos em camadas, como detalhado a seguir.

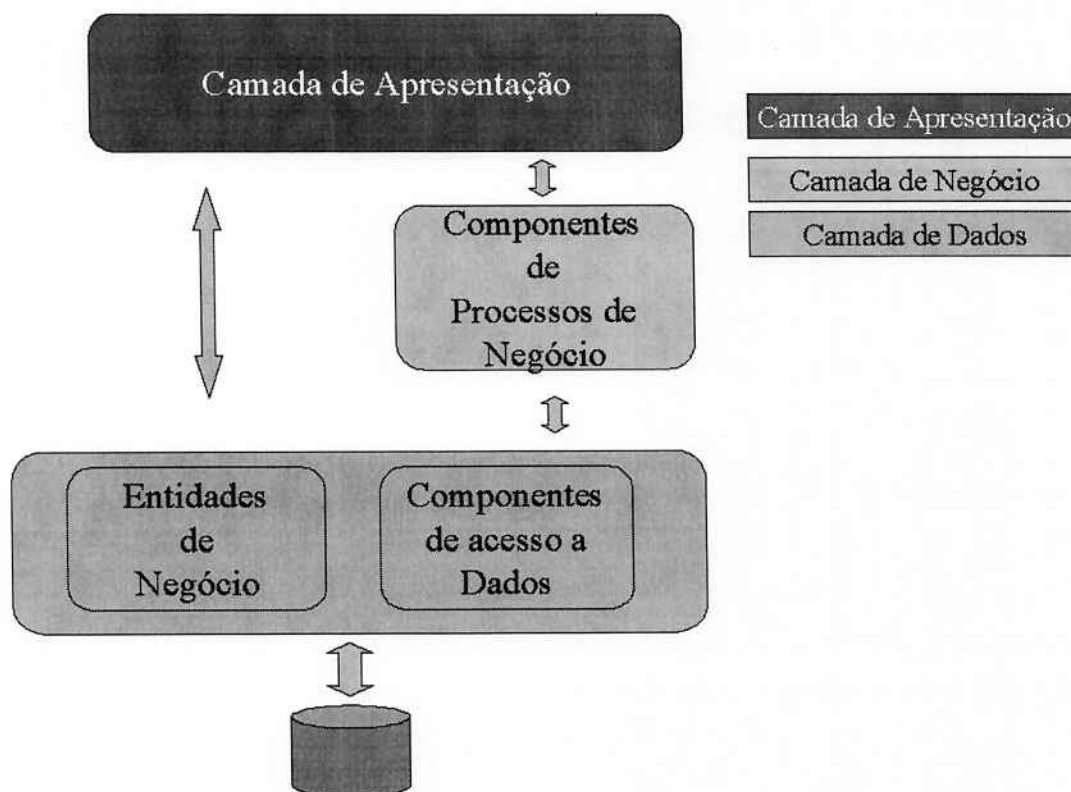


Figura 4 - Camadas de uma aplicação Distribuída

A figura 4 apresenta as camadas mais comuns de uma aplicação distribuída. A seguir, as camadas e alguns de seus componentes são detalhadas.

6.3.1.1. Camada de Apresentação

Esta camada é responsável por fornecer uma interface de acesso ao sistema. Entre as possibilidades de interfaces estão: comandos de texto, janelas de programas ou interfaces Web.

6.3.1.2. Camada de Negócios

Nesta camada estão concentradas todas as regras relacionadas ao domínio da aplicação, necessárias para manter o sistema consistente do ponto de vista do negócio.

6.3.1.3. Camada de Dados

A camada de Dados é a camada responsável por manipular os dados. A escolha do formato de dados, que serão detalhados na seção seguinte, depende da maneira como se pretende trabalhar com os dados. É recomendável que se evite a transferência de dados em formatos personalizados, porque isto requer que se implemente também a serialização de objetos de forma personalizada e isto pode gerar problemas de desempenho. Geralmente, deve-se utilizar um formato centrado nos dados, *DataSet* por exemplo, para transferir dados entre componentes de acesso a dados e as camadas de negócio, podendo-se utilizar os dados contidos neste formato para alimentar classes de negócio personalizadas, caso se queira trabalhar de uma forma mais pura de orientação a objetos. Na maioria dos casos, entretanto, é mais fácil simplesmente utilizar diretamente os dados de negócio contidos no *DataSet*.

6.3.1.4. Componentes de Acesso a Dados

Os componentes de acesso a dados são responsáveis por recuperar e salvar no banco de dados os dados de uma determinada entidade., além de conter a toda lógica de negócio necessária para realizar quaisquer operações que sejam associadas a dados. Permitem tornar a aplicação independente de detalhes do banco de dados como fontes de dados (*data source*), informações de conexão e nomes de colunas de tabelas.

Um componente de acesso a dados oferece métodos que realizam as seguintes operações sobre o banco de dados:

- Criar registros;
- Obter registros, retornando os dados de uma entidade de negócio para quem ativou o método;
- Atualizar registros no banco de dados, utilizando para isto os dados revisados que foram passados como argumento para o método;

- Remover registros do banco de dados.

Os métodos que realizam tais operações são geralmente chamados de “CRUD”, onde “CRUD” é o acrônimo baseado nas primeiras letras das referidas operações (em inglês, CReate, Update, Delete).

Um componente de acesso a dados também implementa métodos de lógica de negócio que manipulam o banco de dados. Por exemplo, um método para encontrar o produto mais vendido em um determinado mês.

Normalmente, um componente de acesso a dados utiliza um único banco de dados e encapsula as operações relacionadas a dados de uma única tabela ou grupo de tabelas relacionadas. Por exemplo, poderia ser definido um componente de acesso a dados para manipular as tabelas de clientes e endereços em um banco de dados, e um outro componente para lidar com as tabelas de pedidos e de detalhamento de pedidos.

Cada Componente de Acesso a Dados manipula um tipo específico de Entidade de Negócio. Por exemplo, o Componente de Acesso a Dados de Clientes lida com as Entidades de Negócio associadas a clientes.

Um Componente de Acesso a Dados é uma classe que não armazena estado (*stateless*), o que significa que todas as trocas de mensagens são interpretadas de forma independente. Nenhum estado é salvo entre diferentes chamadas. O Componente de Acesso a Dados fornece métodos para acessar uma ou mais tabelas de um determinado banco de dados, ou até de vários bancos no caso em que a base de dados é particionada horizontalmente. Tipicamente, os métodos de um Componente de Acesso a Dados chamam *stored procedures* para realizar suas operações.

O Componente de Acesso a Dados trata os seguintes detalhes de implementação:

- Gerencia e encapsula mecanismos de bloqueio de registros;
- Gerencia apropriadamente os aspectos de segurança e autorização;
- Gerencia adequadamente transações;
- Realiza paginação de dados;
- Realiza eventuais roteamentos baseados nos dados;
- Implementa uma estratégia de cache, quando apropriado, para consultas não transacionais;
- Realiza serialização dos dados.

Se uma aplicação contém múltiplos Componentes de Acesso a Dados, pode-se simplificar a implementação das classes desses componentes utilizando-se um componente de apoio de acesso a dados. Tal componente pode ajudar a gerenciar a conexão com o banco de dados, executar comandos SQL e fazer um cache dos parâmetros. Os Componentes de Acesso a Dados ainda encapsulam a lógica necessária para acessar dados específicos de negócio, e o componente de apoio centraliza o código que referencia a API de acesso aos dados e configuração de conexão, ajudando a reduzir a duplicação de código. A Microsoft fornece um componente chamado Data Access Application Block for .NET, que pode ser usado como um componente genérico de apoio, caso a aplicação utilize bancos de dados Microsoft SQL Server™.

Se existirem funções que são comuns a todos os Componentes de Acesso a Dados, pode-se definir uma classe base para todos eles e derivá-la, herdando e estendendo suas características em cada componente específico.

O Componente de Acesso a Dados deve ser projeto de modo a fornecer uma interface consistente para os diferentes tipos de clientes. O desenvolvimento de interfaces consistentes para as camadas de negócio minimiza o risco de se precisar implementar novas interfaces e camadas de mapeamento no futuro.

6.3.1.5. Componentes da Entidade de Negócio

Os dados de um componente da entidade de negócio são usados para representar entidades reais do mundo dos negócios, como por exemplo, produtos e pedidos. Há várias formas de representar entidades de negócio em uma aplicação, como por exemplo: XML, DataSets ou Classes personalizadas.

A seguir, são apresentados as vantagens e desvantagens de algumas opções de implementação das Entidades de Negócio:

- XML: Nesta opção, utiliza-se uma cadeia XML ou um modelo de documento XML no padrão DOM para representar os dados da entidade de negócio. O formato XML é uma representação de dados aberta e flexível que pode ser usada para integrar diversos tipos de aplicações.
- DataSet: Um DataSet é um cache de tabelas em memória, obtido de um banco de dados ou de um documento XML. Um componente de acesso a dados pode usar

um DataSet para representar os dados de uma entidade de negócios recuperados do banco de dados, e pode-se utilizar o próprio DataSet dentro da aplicação.

- **DataSet Tipado:** Um DataSet com tipo é uma classe derivada da classe ADO.NET DataSet e que implementa verificação de tipos para seus métodos, eventos e propriedades que manipulam as tabelas e colunas do DataSet.
- **Componentes de Entidade de Negócio:** Esta opção implementa uma classe personalizada para representar cada entidade de negócio. São definidos os campos que contém os dados da entidade de negócios, e esses dados são expostos através de propriedades que são acessadas pela aplicação. Métodos são implementados para encapsular a lógica de negócio, fazendo uso dos campos definidos na classe. Esta opção não implementa os métodos CRUD para acessar os componentes de acesso a dados, a aplicação cliente se comunica diretamente com os componentes de acesso a dados para realizar as operações de inserção, atualização e remoção de elementos da base de dados.
- **Componentes de Entidade de Negócio com Operações CRUD:** Neste caso, são definidas classes personalizadas, como descrito anteriormente, e são implementados métodos que realizam as operações CRUD chamando o componente de acesso a dados associado à entidade de negócio.

Ao se definir Entidades de Negócio que representam informações na base de dados, deve-se considerar como a informação será usada na aplicação. Deve-se identificar as principais entidades de negócio que encapsulam a funcionalidade da aplicação, ao invés de se definir uma entidade de negócio separada para cada tabela da base. Para cada uma dessas entidades de negócio, um componente de acesso a dados distinto deverá ser definido. Baseado nisso, o que se deve fazer é analisar e modelar as Entidades de Negócio lógicas da aplicação ao invés de definir entidades de negócio separadas para cada tabela.

Não devem ser definidas Entidades de Negócio separadas para representar tabelas de relacionamento “N para N” da base de dados. Estes tipos de relacionamento devem ser expostos através de métodos implementados em um Componente de Acesso a Dados associado àquela Entidade de Negócio.

6.4. Classes

A figura a seguir exibe as classes de negócio do SCE e seus relacionamentos.

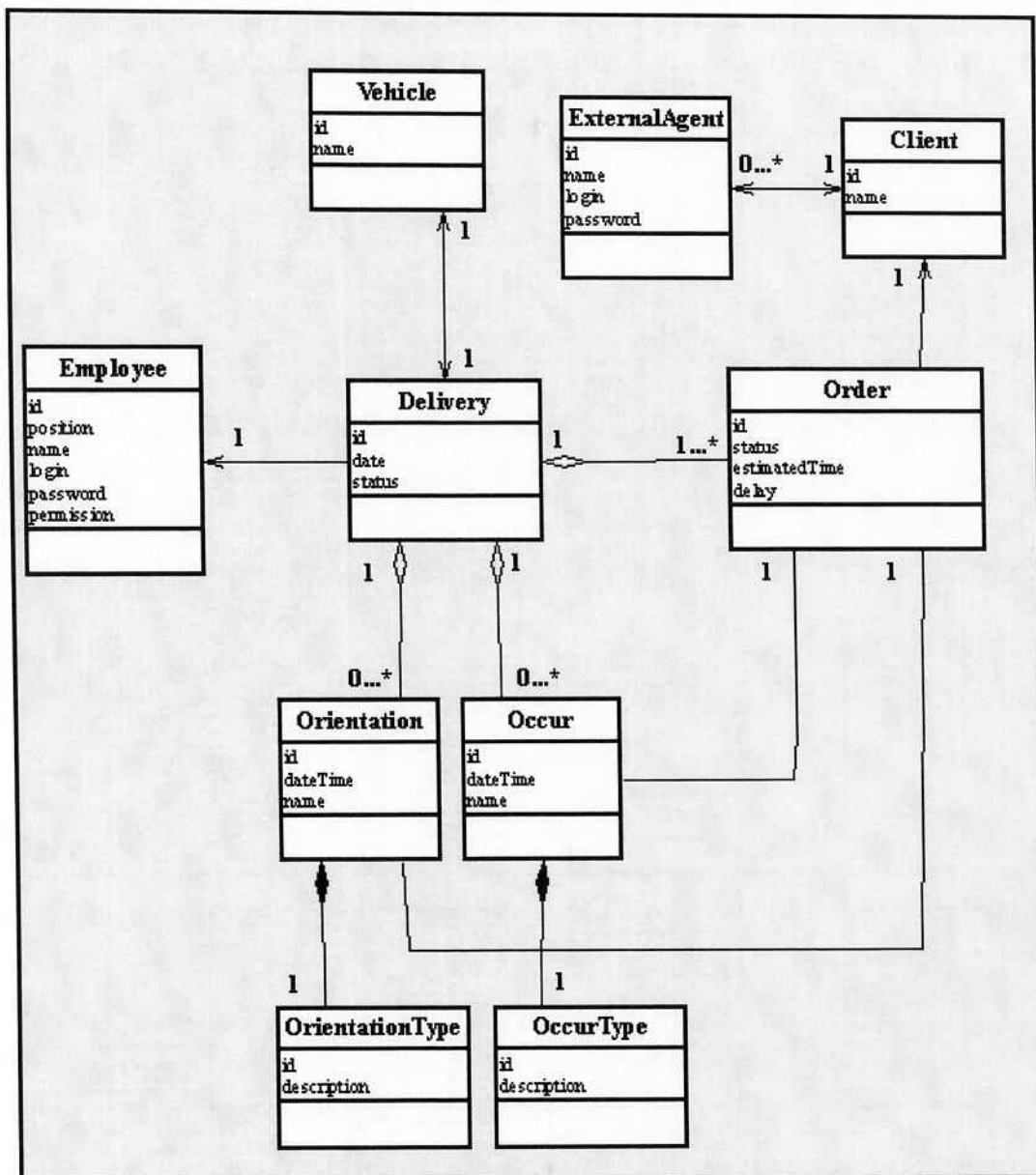


Figura 5 – Diagrama de Classes

A figura 5, representa o diagrama lógico das classes do SCE. Devido ao tipo de arquitetura adotado, conforme detalhado anteriormente, a implementação física não condiz exatamente com a figura.

Nesse projeto, foi escolhido utilizar a solução de Componentes de Entidade de Negócio sem operações CRUD, o que permite uma melhor personalização dos componentes e um modo “mais orientado a objetos” de se trabalhar.

Com isso, as classes foram divididas em 2 tipos: as classes Entidades e as classes de papéis de Negócio, como descritas abaixo:

- Entidades (*Entities*): São os componentes de entidade de negócio, como apresentado anteriormente.
- Papéis de Negócio (*Business Role*): são as classes de acesso a dados. Essas classes contêm os *web methods*, que são métodos que manipulam as *entities*, passando-as para a camada de apresentação ou para a camada de dados. Essas classes contêm os métodos CRUD.

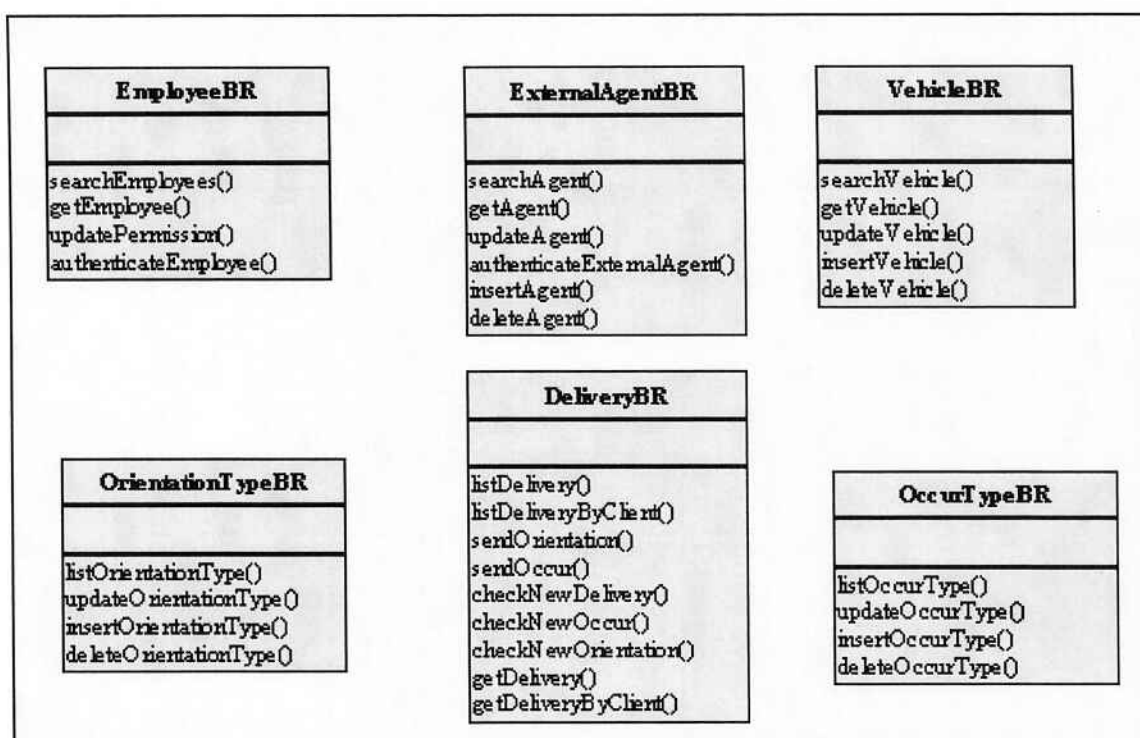


Figura 6 - Diagrama de Classes (Negócio)

A figura 6, refere-se às classes de negócio utilizadas no SCE.

A seguir, são exibidas as classes entidades de negócio.

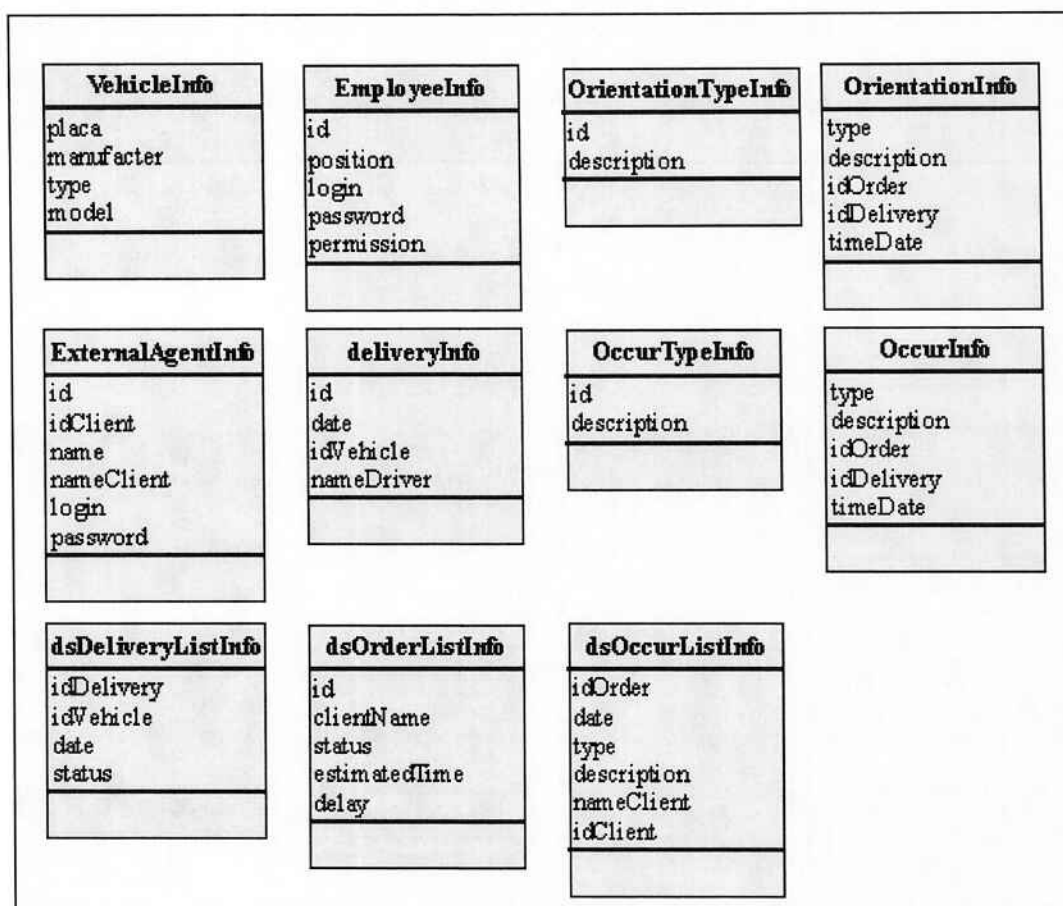


Figura 7 - Classes de Entidade de Negócio

Uma descrição detalhada das classes, encontra-se em anexo [B].

6.5. Interface

A interface administrativa será do estilo WEB. Também haverá uma interface para o POCKET, a ser utilizado pelos motoristas remotamente. Detalhes do projeto da interface, encontram-se em anexo [D].

6.6. Banco de Dados

O Sistema Gerenciador de Banco de Dados a ser utilizado é Microsoft SQL Server. Detalhes sobre o projeto de banco de dados encontra-se em anexo [E].

6.7. Hardware

Nesta seção, serão exibidas informações relativas ao projeto de hardware do SCE.

6.7.1. Arquitetura do SCE

Segue um diagrama ilustrando a arquitetura utilizada no SCE.

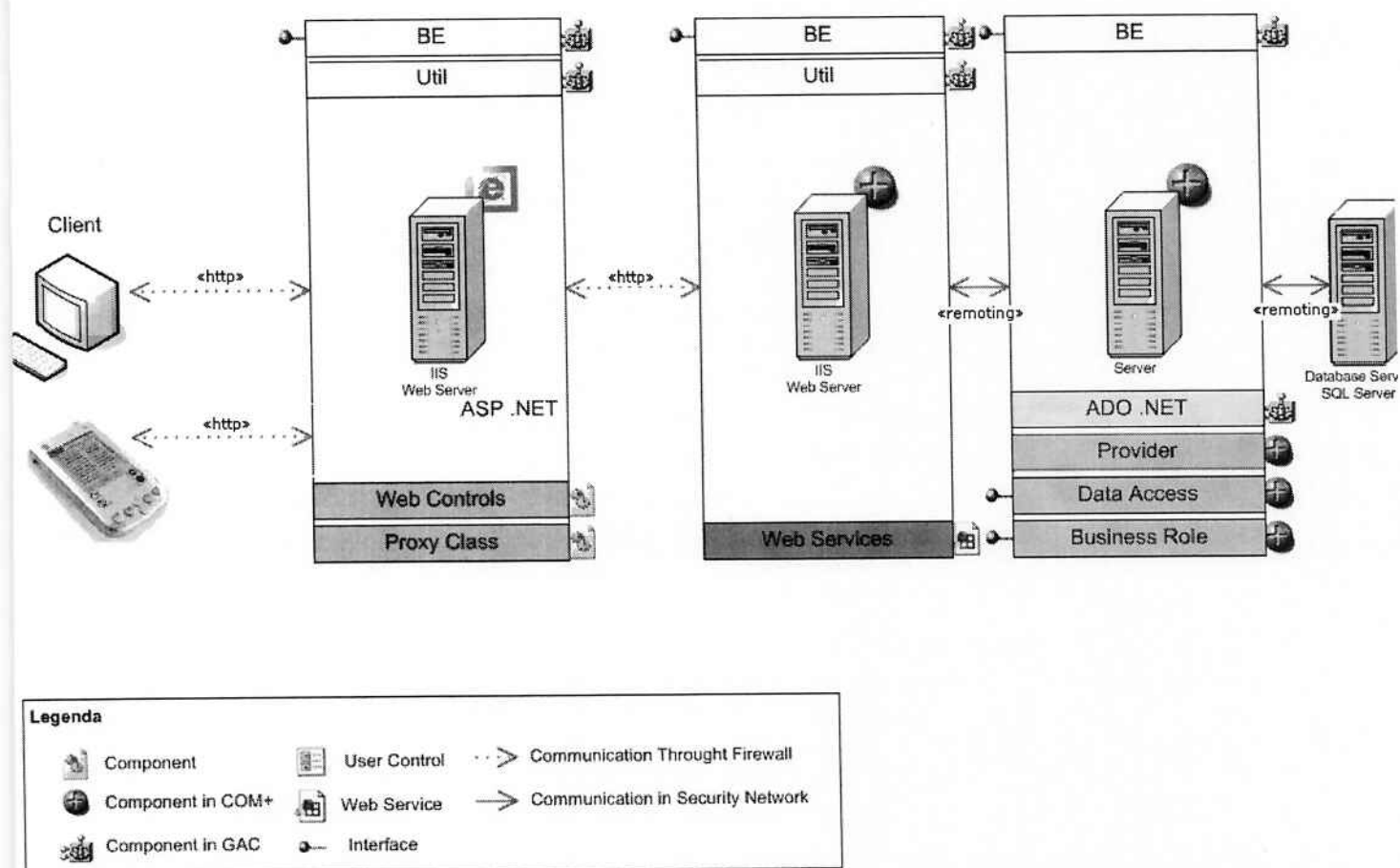


Figura 8 - Arquitetura do Sistema

A figura 8 refere-se à arquitetura de software em hardware do SCE.

Nela pode-se localizar alguns componentes já descritos na seção de software como o ADO, o Data Access, e as classes de negócio (*Business Role*).

De acordo com a figura 8, verifica-se que é possível fazer a implantação dos módulos em máquinas separadas, o que traria uma melhor performance do sistema. Porém, no caso desse projeto, que tem objetivos acadêmicos, todos os componentes estão alocados na mesma máquina por questão de restrições de infra-estrutura e pelo fato de o projeto não requerer grandes níveis de desempenho.

Foi utilizada a arquitetura de componentes distribuídos para a construção do SCE. Esta arquitetura é recomendada para aplicações de grande porte, que requerem alta escalabilidade. Como a proposta do SCE é servir a empresas grandes, o número de acessos simultâneos poderá ser alto, além claro dos acessos dos clientes. Então a Arquitetura de Componentes Distribuídos é a mais indica.

Apesar da arquitetura proposta suportar a utilização de um servidor para cada domínio da aplicação, optou-se por utilizar-se apenas uma máquina para a implantação do SCE.

Baseada nas *best practices* do desenvolvimento orientado a objetos o SCE utiliza alguns Design Patterns na sua implementação.

7. Planejamento e Metodologia

7.1. Metodologia

A metodologia de projeto utilizada foi baseada no modelo cascata. Segundo VENTURA, P., este modelo foi idealizado em 1970 por Royce, e tem como característica principal, a seqüencialidade das atividades, sugerindo um tratamento ordenado e sistemático ao desenvolvimento do software. Cada fase transcorre completamente e seus produtos são vistos como entrada para a nova fase. O software é desenvolvido em um longo processo e entregue ao final deste. Grande parte do sucesso do modelo cascata está no fato dele ser orientado para documentação. O autor sugere laços de *feedback*, que permitem realimentar fases anteriores do processo, mas em geral o modelo cascata é considerado um modelo linear. A figura 95 fornece uma descrição visual do modelo.

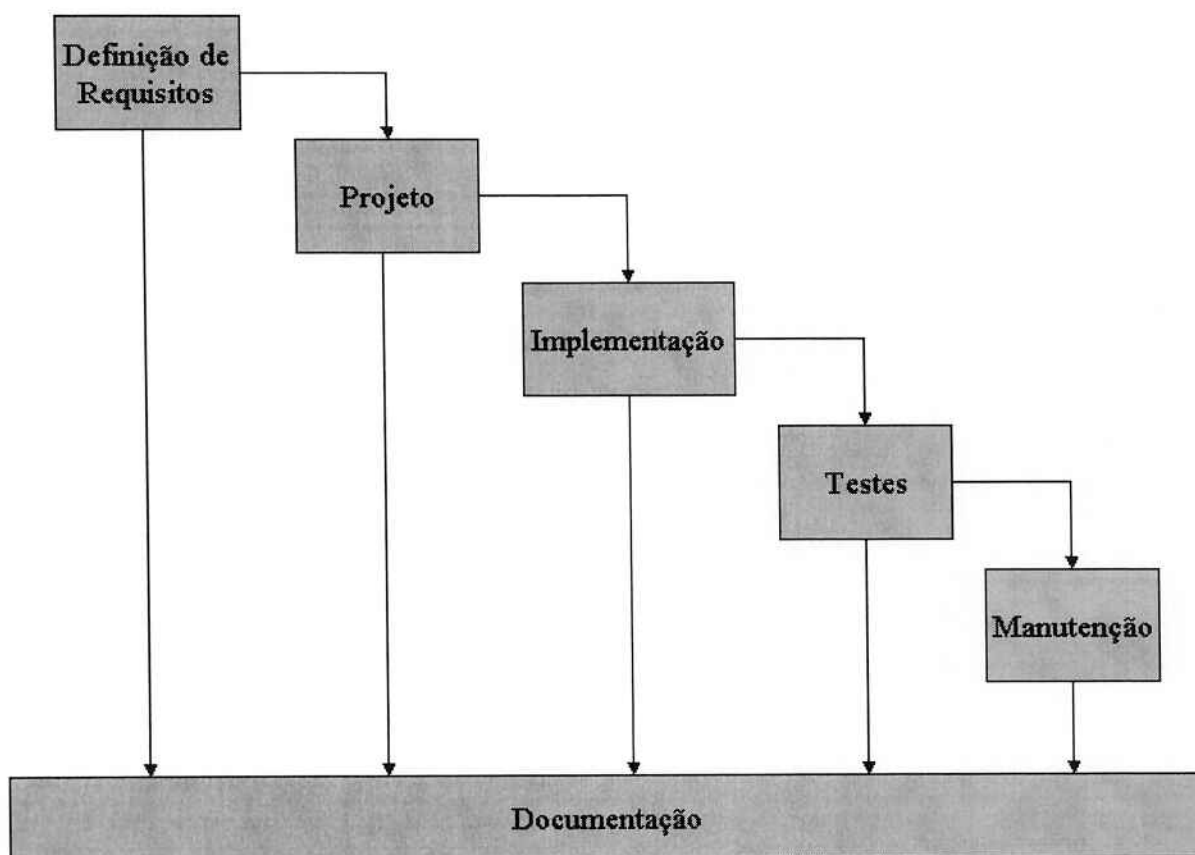


Figura 9 – Modelo Cascata de Projeto de Software

7.2. Etapas do Modelo Cascata

Especificação (definição dos requisitos): Nesta etapa, estabelecem-se os requisitos do produto que se deseja desenvolver, o que consiste usualmente nos serviços que se devem fornecer, limitações e objetivos do software. Sendo isso estabelecido, os requisitos devem ser definidos de uma maneira apropriada para que sejam úteis na etapa seguinte. Toda a especificação é documentada.

Projeto: O projeto do sistema é um processo de vários passos que se centraliza em quatro atributos diferentes do sistema: estrutura de dados, arquitetura do software, detalhes de processos e caracterização das interfaces. O processo de projeto representa os requisitos de uma forma que permita a codificação do produto (é uma prévia etapa de codificação). Da mesma maneira que a análise dos requisitos, o projeto é documentado e transforma-se em uma parte do software.

Implementação: Esta é a etapa em que são criados os programas. Se o projeto possui um nível de detalhe elevado, a etapa de codificação pode implementar-se automaticamente. A princípio, sugere-se incluir um teste unitário dos módulos nesta etapa; nesse caso, as unidades de código produzidas são testadas individualmente antes de passar a etapa de integração e teste global.

Teste do sistema: Concluída a codificação, começa a fase de teste do sistema. O processo de teste centraliza-se em dois pontos principais: as lógicas internas do software e as funcionalidades externas. Esta fase decide se foram solucionados erros de “comportamento” do software e assegura que as entradas definidas produzam resultados reais que coincidam com os requisitos especificados.

Manutenção: Essa etapa consiste na correção de erros que não foram previamente detectados, em melhorias funcionais e de preferência e outros tipos de suporte. A etapa de manutenção é à parte do ciclo de vida do produto de software e não pertence estritamente ao seu desenvolvimento.

7.3. Cronograma

A seguir, estão detalhadas as atividades realizadas durante o projeto.

		Atividade	Início	Fim
1a fase	1	Levantamento das Atividades do Processo de Logística	15-fev-04	20-fev-04
	2	Modelagem do Processo de Transporte de Mercadorias	20-fev-04	28-fev-04
	3	Definição do escopo do sistema	28-fev-04	1-mar-04
	4	Especificação Inicial dos Casos de Uso	3-mar-04	30-abr-04
	5	Especificação da Arquitetura do Sistema	30-abr-04	30-abr-04
	6	Criação do documento preliminar do projeto de formatura	28-mar-04	5-abr-04
Módulo de estágio				
2a fase	7	Revisão Especificação de Casos de Uso	6-set-04	17-set-04
	8	Revisão Especificação de Arquitetura	6-set-04	17-set-04
	9	Construção de Protótipo	18-set-04	1-out-04
	10	Geração da Documentação Final	18-set-04	1-dez-04
	11	Especificação de Classes	18-set-04	18-out-04
	12	Especificação da Interface	18-set-04	18-out-04
	13	Especificação do Banco de Dados	18-out-04	22-out-04
	14	Criação do Plano de Aceitação	25-out-04	27-out-04
	15	Criação do manual do Usuário	28-out-04	3-nov-04
	16	Implementação	28-out-04	1-dez-04
	17	Testes	2-dez-04	8-dez-04

7.4. Aplicação do Modelo Cascata ao Projeto

Comparando-se com o modelo Cascata apresentado anteriormente, percebe-se que as atividades apresentadas no cronograma não condizem totalmente com ele.

A seguir, compara-se a metodologia aplicada no projeto com o Modelo Cascata e a justificativa dessas diferenças.

Como fase de Especificação, pode-se considerar as atividades de 1 a 4, onde levantou-se todo o processo de Logística, definiu-se o escopo e o foco do sistema. Como produtos dessa fase, elaborou-se um diagrama de atividades do processo, a lista de funcionalidades e uma especificação de casos de uso.

Em relação à fase de projeto, pode-se considerar as atividades de 5 a 15. Nela, especificou-se a arquitetura de software, de hardware, as classes, a interface homem-máquina, o banco de dados, além do plano de aceitação. Nessa fase, podemos destacar a atividade construção do protótipo (9), que resultaria em um adiantamento na fase de implementação, pelo fato do esqueleto do sistema já estar montado e o conhecimento sobre a tecnologia maior. Esse conceito foi extraído de projetos baseados na metodologia RUP, onde as atividades são realizadas paralelamente.

As fases de implementação e de Testes, cujo documento descritivo está em anexo [F], não possuem nenhum destaque a ser comentado.

Já a fase de Manutenção, apresentada na seção 7.2 não se aplica ao projeto, pelo fato de ser um projeto acadêmico.

8. CONCLUSÃO

Durante a realização desse trabalho de formatura observou-se que a utilização das tecnologias móveis nos próximos anos poderá ser usada para trazer um diferencial competitivo para as empresas que se propuserem a implantá-las. Nesse trabalho em específico, esse tipo de tecnologia foi utilizada para realizar o controle e a supervisão em tempo real de entregas de mercadorias. Com a implantação de um sistema desse tipo a empresa passaria a utilizar, de uma forma mais racional os recursos envolvidos nesse processo. Isso é alcançado através da análise dos relatórios de desempenho da frota e da equipe de suporte que o sistema fornece.

Além disso, pode-se concluir através desse trabalho que a elaboração de um protótipo do sistema traz um benefício considerável no processo de desenvolvimento. Com a elaboração do protótipo as funções que o sistema deveria desempenhar ficaram mais claras facilitando assim o reconhecimento de incoerências entre os diversos artefatos produzidos nas etapas de desenvolvimento de software.

9. LISTA DE REFERÊNCIAS

AMBLER, S W (2004). *The Diagrams of UML 2.0*

www.agilemodeling.com/essays/umlDiagrams.htm

COMEXNET (2004). Artigo: *Entenda a Importância da Logística*.

www.comexnet.com.br/logistica.htm

GASNIER, D (2004). *Logística não é só transporte*.

www.aesetorial.com.br/transportes/artigos/2004/ago/04/406.htm

MARQUES, V. *Utilizando o TMS (Transportation Management System) para uma Gestão Eficaz de Transportes*. www.cel.coppead.ufrj.br/fs-busca.htm?fr-tms.htm

MICROSOFT. www.microsoft.com

MODELOS DE DOCUMENTAÇÃO DO LABORATÓRIO DE ENGENHARIA DE SOFTWARE DA EPUSP (PCS2034). www.pcs.usp.br/~pcs2034

OMG. www.omg.org

VENTURA, P. *Material do curso de Engenharia de Programação da Faculdade de Ciências de Tecnologia da Universidade do Algarve de Portugal*.

w3.ualg.pt/~pventura/ep/

WIKIPEDIA. en.wikipedia.org

THE WIRELESS LAN BIBLE

www.wirelessbrasil.org/eduardo_prado/wlanbible.html

ANEXO A – ESPECIFICAÇÃO DE CASOS DE USO

1. Gerir Funcionário

1.1. *Listar Funcionários*

Descrição: Lista os funcionários de acordo com as informações fornecidas pelo gerente de transporte.

Evento iniciador: gerente seleciona a opção de listar os funcionários.

Atores: Gerente de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Ator fornece o nome ou o código do funcionário..
2. Sistema exibe a lista de funcionários.

Pós-condição: Sistema exibindo a lista de funcionários.

Extensões:

1. Sistema não encontrou nenhum funcionário de acordo com as informações fornecidas: sistema exibe mensagem de erro indicando que não encontrou nenhum funcionário (passo 1).

1.2. *Consultar Funcionário*

Descrição: Exibe as informações do funcionário selecionado pelo gerente.

Evento iniciador: Seleção de um funcionário específico na lista de funcionários da busca.

Atores: Gerente de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Gerente seleciona um dos funcionários exibidos na lista.
2. SCE exibe as informações referentes àquele funcionário.

Pós-condição: Informações do funcionário sendo exibidas.

1.3. Atribuir permissão de acesso

Descrição: Atribui permissão de acesso ao funcionário selecionado pelo gerente.

Evento iniciador: seleção de um funcionário na lista de busca.

Atores: Gerente de transporte.

Pré-condição: sessão do usuário válida e lista de funcionário sendo exibida.

Seqüência de eventos:

1. O gerente seleciona o funcionário na lista.
2. SCE exibe as informações referentes àquele funcionário.
3. O gerente alterar o cargo desse funcionário.
4. O gerente confirma essa alteração.
5. Sistema exibe uma mensagem confirmando a alteração.

Pós-condição: Funcionário com o cargo alterado.

2. Gerir Agentes Externos

2.1. Inserir Agente externo

Descrição: Adiciona um agente externo para um cliente selecionado.

Evento iniciador: Auxiliar de transporte seleciona a opção para adicionar um novo agente externo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e cliente selecionado.

Seqüência de eventos:

1. O auxiliar de transporte preenche os dados do agente externo.
2. Auxiliar de transporte confirma o cadastro desse novo agente.
3. Sistema exibe uma mensagem indicando se a operação foi corretamente executada.

Pós-condição: Novo agente externo adicionado ao sistema.

Extensões:

1. Agente já cadastrado: sistema exibe mensagem de erro indicando que esse agente já foi cadastrado no sistema (passo 2).
2. Dados do agente externo fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de agente externo (passo 2).

2.2. Listar Agentes externos

Descrição: Lista todos os agentes externos cadastrados para o cliente selecionado.

Evento iniciador: busca de um agente externo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Auxiliar de transporte seleciona o cliente e/ou digita o nome do agente externo desejado.
2. Sistema exibe a lista dos agentes que estão de acordo com a informação fornecida.

Pós-condição: Lista dos agentes externos sendo exibida.

Extensões:

1. Sistema não encontrou nenhum agente externo para esse cliente ou que possua o nome indicado : sistema exibe mensagem de erro e permite que o auxiliar de transporte cadastre um novo agente externo (passo 1).

2.3. Consultar Agente externo

Descrição: Exibe as informações do agente externo selecionado.

Evento iniciador: Auxiliar de transporte seleciona um dos agentes externos exibidos na lista.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo a lista de agentes externos.

Seqüência de eventos:

1. Auxiliar de transporte seleciona um dos agentes externos exibido na lista.
2. Sistema exibe as informações referentes ao agente selecionado.

Pós-condição: Sistema exibindo as informações do agente externo.

2.4. Alterar Agente externo

Descrição: Altera as informações de um agente externo já cadastrado.

Evento iniciador: seleção de um agente externo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo lista de agentes externos.

Seqüência de eventos:

1. Auxiliar de transporte seleciona um dos agentes externos na lista.
2. SCE exibe as informações referentes ao agente externo selecionado.
3. O auxiliar de transporte atualiza as informações desse agente.
4. O auxiliar de transporte confirma essa alteração.
5. SCE exibe uma mensagem indicando se a operação foi corretamente realizada.

Pós-condição: Informações do agente alteradas.

Extensões:

1. Dados do agente externo fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário do agente externo (passo 4).

3. Gerir Veículos

3.1. *Inserir Veículo*

Descrição: Adiciona um novo veículo no sistema.

Evento iniciador: Auxiliar de transporte seleciona a opção para adicionar um novo veículo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. O auxiliar de transporte preenche os dados do veículo.
2. O auxiliar de transporte confirma o cadastro desse novo veículo.
3. Sistema exibe uma mensagem indicando se a operação foi corretamente executada.

Pós-condição: Novo veículo cadastrado no sistema.

Extensões:

1. Veículo já cadastrado: sistema exibe mensagem de erro indicando que já foi cadastrado no sistema um veículo com esses dados (passo 2).
2. Dados do veículo fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de veículo (passo 2).

3.2. Listar Veículos

Descrição: Lista todos os veículos.

Evento iniciador: busca de um veículo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Auxiliar de transporte fornece a placa ou modelo dos veículos que serão exibidos.
2. SCE lista todos os veículos que estão de acordo com as informações fornecidas.

Pós-condição: Sistema exibindo Lista de veículos.

Extensões:

1. Sistema não encontrou nenhum veículo com a descrição indicada: sistema exibe mensagem de erro e permite que o auxiliar de transporte cadastre um novo veículo (passo 1).

3.3. Consultar Veículo

Descrição: Exibe as informações de um veículo selecionado.

Evento iniciador: Auxiliar de transporte seleciona um dos veículos exibidos na lista.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo a lista de veículos.

Seqüência de eventos:

1. Auxiliar de transporte seleciona um dos veículos exibido na lista.
2. Sistema exibe as informações referentes ao veículo selecionado.

Pós-condição: Sistema exibindo as informações do veículo selecionado.

3.4. Alterar Veículo

Descrição: Altera as informações de um veículo já cadastrado.

Evento iniciador: seleção de um veículo.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo lista de veículos.

Seqüência de eventos:

1. Auxiliar de transporte seleciona um dos veículos exibidos na lista.
2. SCE exibe as informações referentes ao veículo selecionado.
3. O auxiliar de transporte atualiza as informações desse veículo.
4. O auxiliar de transporte confirma essa alteração.
5. SCE exibe uma mensagem indicando se a operação foi corretamente realizada.

Pós-condição: Informações do veículo corretamente alteradas.

Extensões:

1. Dados do veículo fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de veículo (passo 4).

4. Gerir Tipos de ocorrências

4.1. *Inserir tipo de ocorrência*

Descrição: Adiciona um novo tipo de ocorrência.

Evento iniciador: Auxiliar de transporte seleciona a opção para adicionar um novo tipo de ocorrência.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. O auxiliar de transporte preenche os dados do novo tipo de ocorrência.
2. O auxiliar de transporte confirma o cadastro.
3. Sistema exibe uma mensagem indicando se a operação foi corretamente executada.

Pós-condição: Novo tipo de ocorrência cadastrado no sistema.

Extensões:

1. Tipo de ocorrência já cadastrado: sistema exibe mensagem de erro indicando que esse tipo de ocorrência já foi cadastrado no sistema (passo 2).
2. Dados do tipo de ocorrência fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de tipo de ocorrência (passo 2).

4.2. Listar tipos de ocorrência

Descrição: Lista todos os tipos de ocorrência.

Evento iniciador: busca de um tipo de ocorrência.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

3. SCE exibe a lista com todos os tipos de ocorrência disponível no sistema.

Pós-condição: Sistema exibindo Lista de tipos de ocorrência.

Extensões:

2. Sistema não encontrou nenhum tipo de ocorrência cadastrado no sistema: sistema exibe mensagem de erro e permite que o auxiliar de transporte cadastre um novo tipo de ocorrência (passo 1).

4.3. Alterar tipo de ocorrência

Descrição: Altera as informações de um tipo de ocorrência já cadastrado.

Evento iniciador: seleção de um tipo de ocorrência.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo lista de tipo de ocorrência.

Seqüência de eventos:

1. Auxiliar de transporte seleciona um dos tipos de ocorrência exibido na lista.
2. SCE exibe as informações referentes ao tipo de ocorrência selecionado.
3. O auxiliar de transporte atualiza as informações desse tipo de ocorrência.
4. O auxiliar de transporte confirma essa alteração.
5. SCE exibe uma mensagem indicando se a operação foi corretamente realizada.

Pós-condição: Informações do tipo de ocorrência corretamente alteradas.

Extensões:

1. Dados do tipo de ocorrência fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de tipo de ocorrência (passo 4).

5. Gerir Tipos de orientações

5.1. Inserir tipo de orientação

Descrição: Adiciona um novo tipo de orientação.

Evento iniciador: Auxiliar de transporte seleciona a opção para adicionar um novo tipo de orientação.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. O auxiliar de transporte preenche os dados do novo tipo de orientação.
2. O auxiliar de transporte confirma o cadastro.
3. Sistema exibe uma mensagem indicando se a operação foi corretamente executada.

Pós-condição: Novo tipo de orientação cadastrado no sistema.

Extensões:

1. Tipo de orientação já cadastrado: sistema exibe mensagem de erro indicando que esse tipo de ocorrência já foi cadastrado no sistema (passo 2).
2. Dados do tipo de ocorrência fornecido não são consistentes: Sistema exibe mensagem de erro e retorna para o formulário de tipo de ocorrência (passo 2).

5.2. Listar tipos de orientação

Descrição: Lista todos os tipos de orientação.

Evento iniciador: busca de um tipo de orientação.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. SCE exibe a lista com todos os tipos de orientação disponível no sistema.

Pós-condição: Sistema exibindo Lista de tipos de orientação.

5.3. Alterar tipo de orientação

Descrição: Altera as informações de um tipo de orientação já cadastrado.

Evento iniciador: seleção de um tipo de orientação.

Atores: Auxiliar de transporte.

Pré-condição: sessão do usuário válida e sistema exibindo lista de tipo de orientação.

Seqüência de eventos:

1. Usuário seleciona um dos tipos de orientação exibido na lista.
2. SCE exhibe as informações referentes ao tipo de orientação selecionado.
3. O ator atualiza as informações desse tipo de orientação.
4. O ator confirma essa alteração.
5. SCE exhibe uma mensagem indicando se a operação foi corretamente realizada.

Pós-condição: Informações do tipo de orientação corretamente alteradas.

Extensões:

1. Dados do tipo de orientação fornecido não são consistentes: Sistema exhibe mensagem de erro e retorna para o formulário de tipo de ocorrência (passo 4).

6. Gerir Entregas

6.1. *Listar Entregas*

Descrição: Lista todas as entregas em andamento.

Evento iniciador: usuário seleciona a lista de entregas.

Atores: Gerente de transporte ou auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Usuário seleciona a lista de entregas.
2. SCE exhibe a lista de entregas em andamento.

Pós-condição: Sistema exibindo Lista de entregas.

6.2. *Consultar Entrega*

Descrição: Exhibe as informações de uma entrega.

Evento iniciador: usuário seleciona uma das entregas da lista.

Atores: Gerente de transporte ou auxiliar de transporte.

Pré-condição: sessão do usuário válida e lista de entregas sendo exibida.

Seqüência de eventos:

1. Usuário seleciona um das entregas da lista.
2. SCE exhibe as informações dessa entrega.

Pós-condição: Sistema exibindo informações da entrega selecionada.

6.3. Consultar Pedido

Descrição: Exibe informações de um pedido.

Evento iniciador: usuário seleciona um pedido.

Atores: Gerente de transporte ou auxiliar de transporte.

Pré-condição: sessão do usuário válida e lista de entregas sendo exibida.

Seqüência de eventos:

1. Ator seleciona um pedido da lista de pedidos de uma entrega.
2. SCE exhibe as informações desse pedido selecionado.

Pós-condição: Sistema exibindo informações do pedido selecionado.

6.4. Enviar Orientação

Descrição: Envio de orientações para o motorista.

Evento iniciador: usuário seleciona a opção de envio de orientação.

Atores: Gerente de transporte ou auxiliar de transporte.

Pré-condição: sessão do usuário válida e lista de entregas sendo exibida.

Seqüência de eventos:

1. Ator preenche os dados da orientação.
2. Ator confirma o envio.
3. SCE envia a orientação e adiciona no histórico de entrega.

Pós-condição: Orientação enviada para o motorista.

7. Verificar Novos Pedidos

Descrição: Verifica se foram enviados novos pedidos pelo sistema de expedição.

Evento iniciador: Pedido enviado pelo sistema de expedição.

Atores: Cron

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Ator verifica se existem novos pedidos.
2. Sistema de expedição envia informações dos pedidos ao SCE.

Pós-condição: pedidos cadastrados no SCE.

8. Verificar chegada de Novas Ocorrências

Descrição: Verificar se existem novas ocorrências na caixa de entrada.

Evento iniciador: seleção do espetáculo desejado

Atores: Cron

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Ator verificar se existem novas ocorrências.
2. Sistema alerta o usuário que existe novas ocorrências.

Pós-condição: Novas ocorrências exibidas pelo sistema.

9. Visualizar Entregas

9.1. Listar Entregas

Descrição: Lista entregas em andamento que contenha pedidos do cliente selecionado.

Evento iniciador: usuário seleciona um pedido.

Atores: Gerente de transporte ou auxiliar de transporte.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Ator fornece o cliente.
2. SCE exhibe a lista de entregas em andamento para esse cliente selecionado.

Pós-condição: Sistema exibindo a lista de entregas desse cliente.

9.2. Consultar Entregas

Descrição: Lista entregas em andamento que contenha pedidos do cliente ao qual esse agente externo. pertence.

Evento iniciador: Agente externo seleciona a lista de entregas.

Atores: Agente externo.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. SCE exibe a lista de entregas em andamento referente ao cliente ao qual o agente externo pertence.

Pós-condição: Sistema exibindo a lista de entregas em andamento.

9.3. Consultar Pedido

Descrição: Lista entregas em andamento que contenha pedidos do cliente ao qual esse agente externo.

Evento iniciador: Agente seleciona um pedido da lista de entrega.

Atores: Agente externo.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Agente externo seleciona um pedido da lista de pedidos de uma entrega.
2. SCE exibe as informações desse pedido.

Pós-condição: Sistema exibindo a informações desse pedido.

9.4. Enviar Ocorrências

Descrição: Envia ocorrência referente à uma entrega. Essa operação poderá ser efetuada na interface administrativa, na interface do agente externo, na interface WAP ou através do sistema (Cron).

Evento iniciador: Usuário seleciona a opção de envio de nova ocorrência.

Atores: Agente Externo, Motorista, Gerente de Transporte, Auxiliar de Transporte e Cron.

Pré-condição: sessão do usuário válida.

Seqüência de eventos:

1. Usuário preenche dados da ocorrência.
2. Sistema registra informações.

Pós-condição: ocorrência cadastrada no sistema.

ANEXO B – MODELO DE CLASSES

10. Entities

10.1. VehicleInfo

Descrição: dados relativos ao cadastro de um veículo

Atributos:

- id
- manufacturer
- type
- model

10.2. EmployeeInfo

Descrição: dados relativos ao cadastro de funcionários

Atributos:

- id
- position
- login
- password
- permission

10.3. ExternalAgentInfo

Descrição: dados relativos ao cadastro de agentes externos

Atributos:

- id
- name
- idClient
- nameClient
- login
- password

10.4. *OrientationTypeInfo*

Descrição: dados relativos ao cadastro de tipos de orientação

Atributos:

- id
- description

10.5. *OccurTypeInfo*

Descrição: dados relativos ao cadastro de tipos de ocorrência

Atributos:

- id
- description

10.6. *OrientationInfo*

Descrição: dados relativos a uma orientação

Atributos:

- date
- idDelivery
- idOrder
- type
- description

10.7. *OccurInfo*

Descrição: dados relativos a uma ocorrência

Atributos:

- date
- idDelivery
- idOrder
- type
- description

10.8. *dsDeliveryListInfo*

Descrição: dataset com lista de entregas

Atributos:

- date
- idDelivery
- idVehicle
- status

10.9. *DeliveryInfo*

Descrição: dados relativos a uma entrega

Atributos:

- id
- date
- idVehicle
- nameDriver

10.10. *dsOrderListInfo*

Descrição: dataset com lista de pedidos de um entrega

Atributos:

- id
- clientName
- status
- estimatedTime
- delay

10.11. *dsOccurListInfo*

Descrição: dataset com lista de ocorrências e orientações raleativos a uma entrega

Atributos:

- date
- idOrder
- type
- description
- nameClient

11. Business Role

11.1. EmployeeBR

Descrição: serviços relativos aos funcionários

Métodos:

- SearchEmployees

Parâmetros	str name, str cargo
Retorno	Dataset
Descrição	Procura funcionários de acordo com as informações passadas como parâmetro

- GetEmployee

Parâmetros	int id
Retorno	EmployeeInfo
Descrição	Retorna informações do funcionário passado como parâmetro

- UpdatePermission

Parâmetros	int id, int permission
Retorno	Void
Descrição	Altera permissão do funcionário passado como parâmetro

- AuthenticateEmployee

Parâmetros	str login, str password
Retorno	Boolean
Descrição	Valida login e senha

11.2. ExternalAgentBR

Descrição: serviços relativos aos agentes externos

Métodos:

- SearchAgent

Parâmetros	str name, id client
Retorno	Dataset
Descrição	Procura agentes externos de acordo com as informações passadas

	como parâmetro
--	----------------

- GetAgent

Parâmetros	int id
Retorno	ExternalAgentInfo
Descrição	Retorna informações do agente externo passado como parâmetro

- UpdateAgent

Parâmetros	ExternalAgentInfo eai
Retorno	Void
Descrição	Altera dados do agente externo passado como parâmetro

- AuthenticateAgent

Parâmetros	Str login, str password
Retorno	Boolean
Descrição	Valida login e senha

- InsertAgent

Parâmetros	ExternalAgentInfo eai
Retorno	Void
Descrição	Insere agente externo passado como parâmetro

- DeleteAgent

Parâmetros	int id
Retorno	Void
Descrição	Exclui agente externo passado como parâmetro

11.3. VehicleBR

Descrição: serviços relativos aos veículos

Métodos:

- SearchVehicle

Parâmetros	str placa, str manufacturer, str type, str model
Retorno	Dataset
Descrição	Procura veículos de acordo com as informações passadas como parâmetro

- GetVehicle

Parâmetros	int id
Retorno	VehicleInfo
Descrição	Retorna informações do veículo passado como parâmetro

- UpdateVehicle

Parâmetros	VehicleInfo vi
Retorno	Void
Descrição	Altera dados do veículo passado como parâmetro

- InsertVehicle

Parâmetros	VehicleInfo vi
Retorno	Void
Descrição	Insere veículo passado como parâmetro

- DeleteVehicle

Parâmetros	int id
Retorno	Void
Descrição	Exclui veículo passado como parâmetro

11.4. OrientationTypeBR

Descrição: serviços relativos aos tipos de orientação

Métodos:

- ListOrientationTypes

Parâmetros	-
Retorno	Dataset
Descrição	Lista tipos de orientação

- UpdateOrientationType

Parâmetros	OrientationTypeInfo oti
Retorno	Void
Descrição	Altera dados do tipo de orientação passado como parâmetro

- InsertOrientationType

Parâmetros	OrientationTypeInfo oti
-------------------	-------------------------

Retorno	Void
Descrição	Insere tipo de orientação passado como parâmetro

- DeleteOrientationType

Parâmetros	int id
Retorno	Void
Descrição	Exclui tipo de orientação passado como parâmetro

11.5. OccurBR

Descrição: serviços relativos aos tipos de ocorrência

Métodos:

- ListOccurs

Parâmetros	-
Retorno	Dataset
Descrição	Lista tipos de ocorrência

- UpdateOccurType

Parâmetros	OccurInfo oti
Retorno	Void
Descrição	Altera dados do tipo de ocorrência passado como parâmetro

- InsertOccurType

Parâmetros	OccurInfo oti
Retorno	Void
Descrição	Insere tipo de ocorrência passado como parâmetro

- DeleteOccurType

Parâmetros	int id
Retorno	Void
Descrição	Exclui tipo de ocorrência passado como parâmetro

11.6. DeliveryBR

Descrição: serviços relativos às entregas

Métodos:

- ListDeliveries

Parâmetros	Int status
Retorno	DsDeliveryListInfo
Descrição	Lista entregas de acordo com o status passado como parâmetro (em andamento ou finalizada)

- ListDeliveriesByClient

Parâmetros	Int status, int idClient
Retorno	DsDeliveryListInfo
Descrição	Lista entregas de acordo com o status (em andamento ou finalizada) e o cliente passado como parâmetro

- SendOrientation

Parâmetros	OrientationInfo oi
Retorno	Void
Descrição	Envia orientação

- SendOccur

Parâmetros	OccurInfo oi
Retorno	Void
Descrição	Envia ocorrência

- CheckNewDelivery

Parâmetros	-
Retorno	Boolean
Descrição	Verifica se existe uma nova entrega na lista de entregas

- CheckNewOccur

Parâmetros	-
Retorno	Boolean
Descrição	Verifica se existe uma nova ocorrência na lista de ocorrências

- CheckNewOrientation

Parâmetros	-
Retorno	Boolean
Descrição	Verifica se existe uma nova orientação na lista de orientações

- GetDelivery

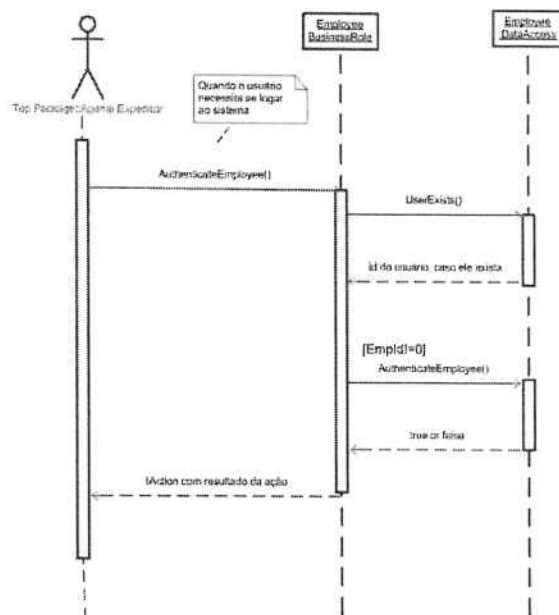
Parâmetros	int id
Retorno	DeliveryInfo
Descrição	Retorna informações da entrega passada como parâmetro

- GetDeliveryByClient

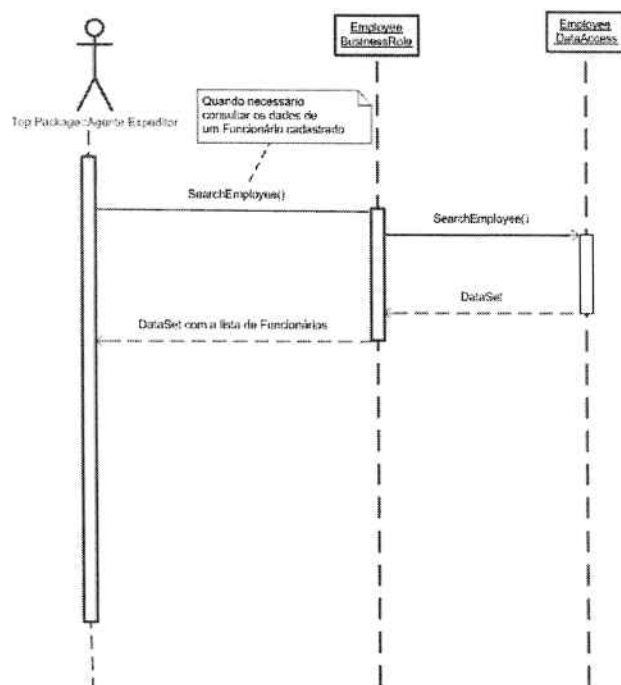
Parâmetros	int id, int idClient
Retorno	DeliveryInfo
Descrição	Retorna informações, relativas ao cliente passado como parâmetro, da entrega passada como parâmetro

ANEXO C – MODELO DINÂMICO

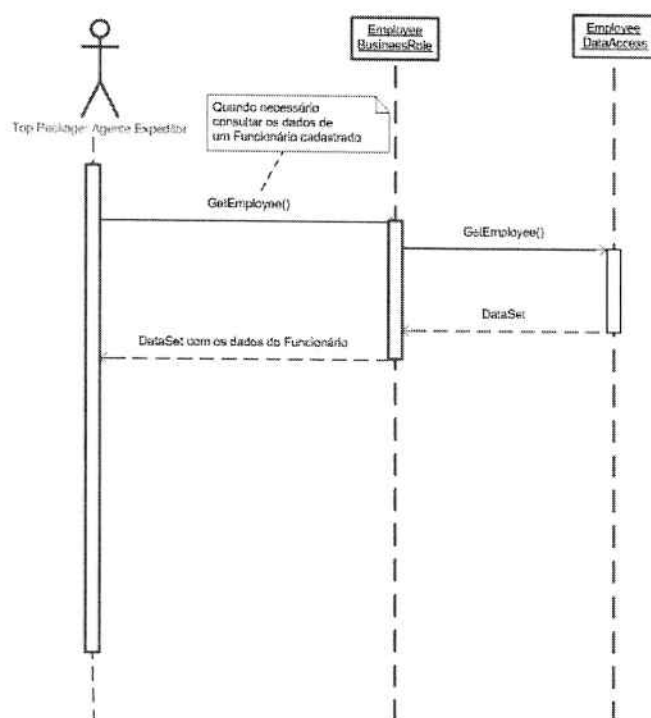
Realizar Login



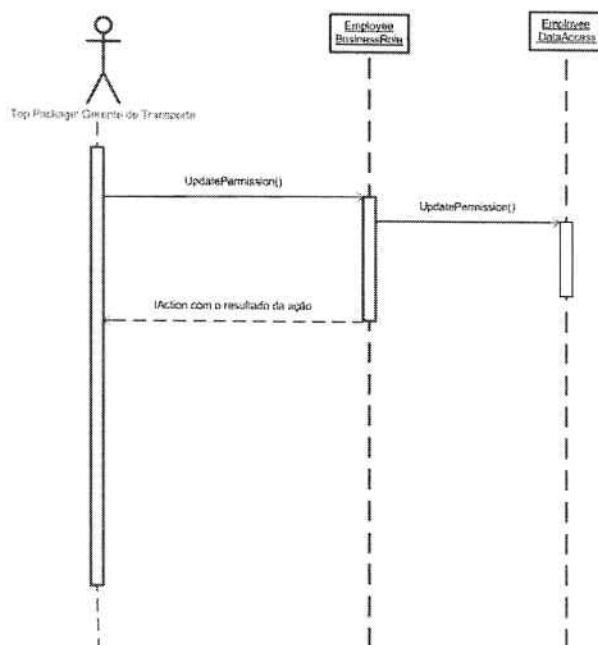
Listar Funcionários



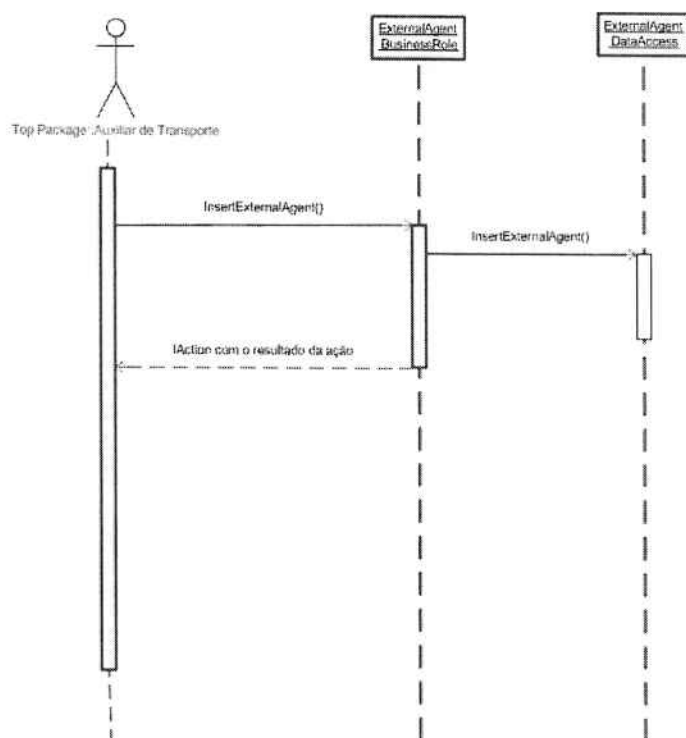
Consultar Funcionário



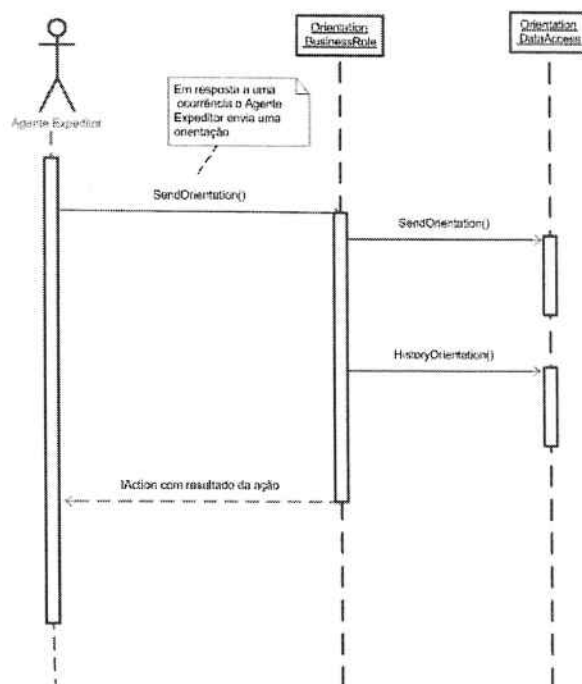
Atribuir permissão de acesso



Inserir Agente Externo



Enviar Orientação



ANEXO D – PROJETO DA INTERFACE DE USUÁRIO

1. Caracterização do Usuário

N	Papel do Usuário	Funções permitidas	Frequência de uso	Conhecimento da tarefa
1	Gerente	Todas	Diária	Alto
2	Auxiliar	Todas, exceto relatórios	Diária	Alto
3	Agente Externo	Adicionar Ocorrências, Listar Entregas (apenas as com pedidos relativos a sua empresa)	Diária	Operacional
4	Motorista	Adicionar Ocorrências	Diária	Operacional

2. Estilo da Interface

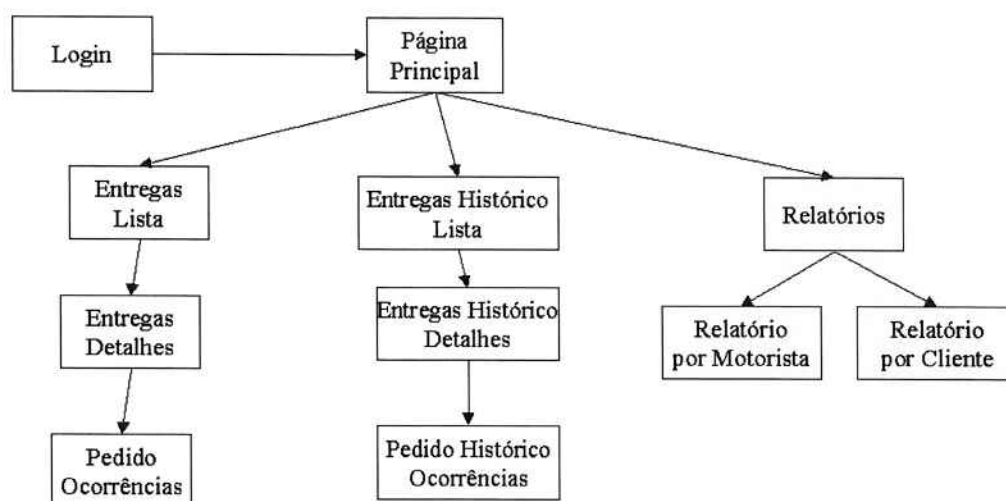
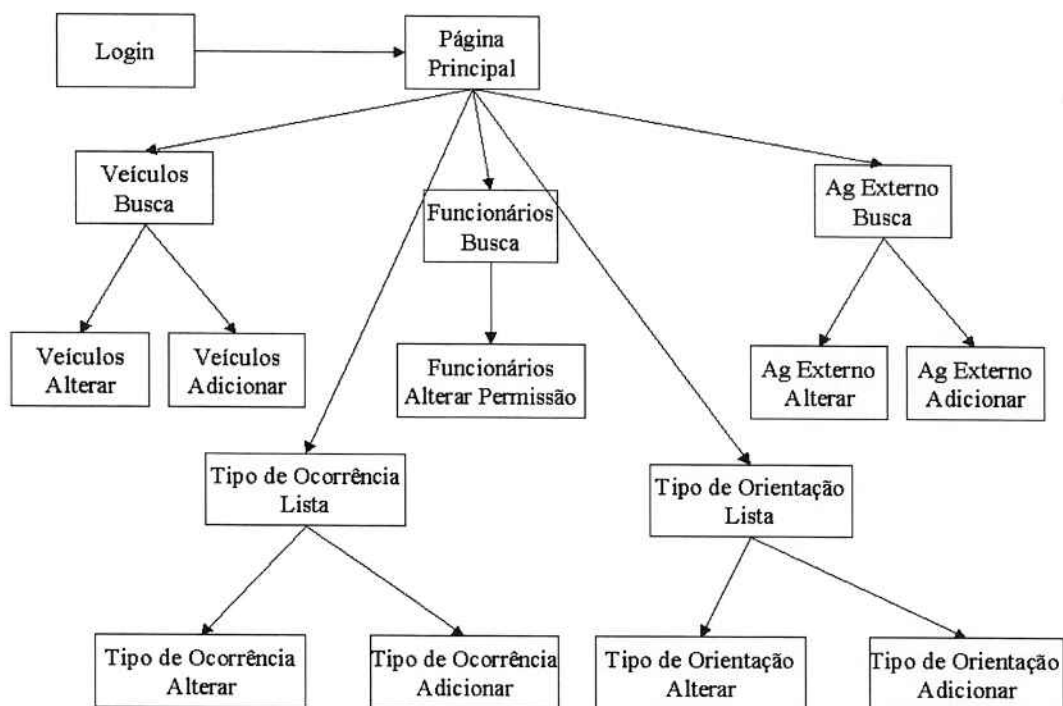
A interface administrativa será do estilo WEB. Também haverá uma interface para o POCKET, a ser utilizado pelos motoristas remotamente.

3. Protótipo de Navegação

A seguir, detalha-se o esquema de navegação da interface do SCE

3.1. Diagrama Hierárquico das Páginas

Figura 10 – Esquema Hierárquico das Páginas



Através da figura 10, podemos visualizar a hierarquia das páginas da interface do SCE. Adotaremos como nível 1 as páginas logo abaixo da página principal, de nível 2 as páginas logo abaixo das de nível 1 e assim por diante. Os nós indicam as páginas e as setas indicam o caminho que deve ser feito através dos links para se alcançar as páginas mais abaixo..

Os nós de nível 1 podem ser alcançados diretamente através de qualquer página através do menu lateral esquerdo, mas os nós dos outros níveis, só podem ser alcançados seguindo o caminho proposto na figura 6. Para esses nós, existem links que levam a última página visitada, fazendo com que o usuários não tenha que voltar para a primeira página quando estiver em uma de nível 3 querendo ir para uma de nível 2.

A figura a seguir, ilustra o layout da interface.

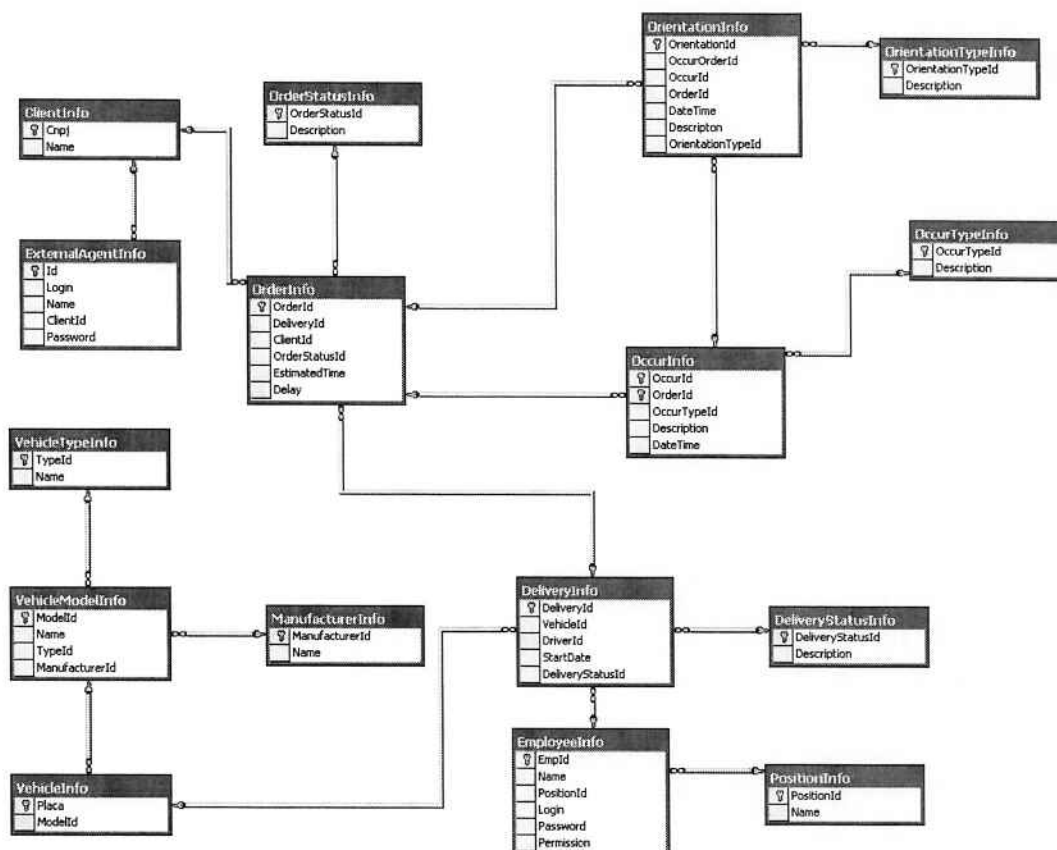
SISTEMA DE CONTROLE DE ENTREGAS																								
Veículo Busca Funcionário Busca Agente Externo Busca Tipo de Ocorrência Lista Tipo de Orientação Lista Entrega Lista Histórico Relatórios	PEDIDO 976534 - OCORRÊNCIAS																							
	Cliente: <input type="text"/>		Status: <input type="text"/>																					
	Previsão de Entrega Original: <input type="text"/>																							
	Atraso Estimado: <input type="text"/>																							
	Ocorrências:																							
	<table border="1"> <thead> <tr> <th>Data</th> <th>Tipo</th> <th>Cod / Descrição</th> <th>Cliente</th> </tr> </thead> <tbody> <tr> <td>01/03/04 - 13:00:00</td> <td>OC</td> <td>2- Início Entrega</td> <td></td> </tr> <tr> <td>01/03/04 - 13:40:00</td> <td>OC</td> <td>4 - Pedido 9393 Entregue</td> <td>Lojas Joao</td> </tr> <tr> <td>01/03/04 - 13:55:05</td> <td>OC</td> <td>7 - Atraso 30 min (trânsito)</td> <td></td> </tr> <tr> <td>01/03/04 - 14:05:05</td> <td>OR</td> <td>2 - Prosseguir próximo Cliente</td> <td></td> </tr> </tbody> </table>	Data	Tipo	Cod / Descrição	Cliente	01/03/04 - 13:00:00	OC	2- Início Entrega		01/03/04 - 13:40:00	OC	4 - Pedido 9393 Entregue	Lojas Joao	01/03/04 - 13:55:05	OC	7 - Atraso 30 min (trânsito)		01/03/04 - 14:05:05	OR	2 - Prosseguir próximo Cliente				
	Data	Tipo	Cod / Descrição	Cliente																				
	01/03/04 - 13:00:00	OC	2- Início Entrega																					
	01/03/04 - 13:40:00	OC	4 - Pedido 9393 Entregue	Lojas Joao																				
	01/03/04 - 13:55:05	OC	7 - Atraso 30 min (trânsito)																					
01/03/04 - 14:05:05	OR	2 - Prosseguir próximo Cliente																						
<div>Voltar</div>																								

Figura 11 – Layout da Interface

ANEXO E – PROJETO DE BANCO DE DADOS

O objetivo deste documento é descrever o projeto de banco de dados.

1. Tabelas do Modelo Relacional



ANEXO F – PLANO DE ACEITAÇÃO

1. Objetivo do Documento

O objetivo do documento de Plano de Aceitação é estabelecer os casos de teste que a serem executados para aceitação do sistema. Devem estar contemplados os testes dos requisitos funcionais e não funcionais, acompanhando-se o Documento de Especificação de Requisitos de Software.

2. Resumo do teste de aceitação

<i>Produto avaliado</i> SCE (Sistema de Controle de Entregas)		
<i>Data</i> 5/12/2004	<i>Hora de início</i> 14:00	<i>Hora de término</i> 16:00

<i>Participantes</i> 1. Daniel Konishi 2. Fabio Luiz Perrella 3. Gilson Alves
--

<i>Resultado do teste de aceitação</i>	
ACEITO <u>OK</u> Como está <u> </u> Com revisões menores	NÃO ACEITO (Nova aceitação é necessária) <u> </u> Revisões maiores <u> </u> Reconstruir <u> </u> Aceitação incompleta

3. Infra-estrutura para a Aceitação

<i>Configuração dos Equipamentos</i>	<i>Software necessário</i>
Pentium III 800 Mhz 512 MB de RAM, 8 GB	Windows 2000 Server ou Windows XP IIS SQL Server 2000

4. Lista de Testes

Segue a lista com os testes a serem realizados.

Teste no.	Tipo	Requisito testado
1	Funcional	Gerir funcionários
2	Funcional	Gerir Agentes Externos
3	Funcional	Gerir Veículos
4	Funcional	Gerir tipos de orientação
5	Funcional	Gerir tipos de ocorrência
6	Funcional	Gerir Entregas
7	Funcional	Envio de Ocorrências
8	Funcional	Visualiza entrega (Agente Externo)
9	Funcional	Controle de acesso

5. Testes

Seguem detalhamento dos testes realizados.

Teste no. 01		Requisito testado: Gerir Funcionário	
Funcional: <input checked="" type="checkbox"/>	Observações: Como o nosso sistema não realiza um cadastro de funcionários independente ele deverá consultar o cadastro de funcionário do sistema de BACK-END		
Não funcional: <input type="checkbox"/>			
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolher a opção para listar funcionário.		O sistema deve exibir uma lista com todos os funcionários do sistema de BACK-END.	ok
2. Escolher a opção de consulta de funcionário.		O sistema deve exibir as informações detalhadas do funcionário selecionado.	ok
3. Atribuir a permissão de acesso ao funcionário selecionado.		O sistema deverá armazenar a permissão de acesso dada desse funcionário.	ok
4. O funcionário escolhido deverá entrar no sistema.		O sistema deverá permitir que esse funcionário entre no sistema e deverá habilitar as ações que ele pode executar de acordo como a permissão dada.	ok
<input checked="" type="checkbox"/> Aprovado	Comentários		
<input type="checkbox"/> Reprovado			
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 02		Requisito testado: Gerir agente externo	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações:		
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolher a opção para listar agente externo.		O sistema deve exibir uma lista com todos os agentes externos do sistema.	Ok
2. Escolher a opção de adicionar um novo agente externo.		O sistema deve exibir o formulário de cadastro de agente externo.	Ok
3. Escolher a opção de listar agente externo.		O sistema exibir a lista de agentes externos como o novo agente externo cadastrado.	Ok
4. Escolher a opção para consultar o agente externo previamente cadastrado.		O sistema deverá exibir as informações que foram cadastradas para esse agente externo.	Ok
5. Escolher a opção para alterar os dados do agente externo previamente cadastrado.		O sistema deve exibir o formulário para alteração as informações desse agente externo.	Ok
6. Escolher a opção para consultar o agente externo previamente cadastrado.		O sistema deve exibir as informações já atualizadas do agente externo.	Ok
Seqüência para verificação de exceções		Comportamento esperado do software	
1. Realizar o teste 1 escolhendo um cliente que não possui agente externo cadastrado		O sistema deverá indicar que não há agentes externos para esse cliente	Ok
<input checked="" type="checkbox"/> Aprovado	Comentários		
<input type="checkbox"/> Reprovado			
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 03		Requisito testado: Gerir veículos	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações:		
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolher a opção para listar os veículos cadastrados.		O sistema deve exibir uma lista com todos os veículos cadastrados no sistema.	Ok
2. Escolher a opção de adicionar um novo veículo.		O sistema deve exibir o formulário de cadastro de veículo.	Ok
3. Escolher a opção de listar veículo.		O sistema exibir a lista de veículos como o novo veículo cadastrado.	Ok
4. Escolher a opção para consultar o veículo previamente cadastrado.		O sistema deverá exibir as informações que foram cadastradas para esse veículo.	Ok
5. Escolher a opção para alterar os dados do veículo previamente cadastrado.		O sistema deve exibir o formulário para alteração as informações desse veículo.	Ok
6. Escolher a opção para consultar o veículo previamente cadastrado.		O sistema deve exibir as informações já atualizadas do veículo.	Ok
7			
Seqüência para verificação de exceções		Comportamento esperado do software	
1. Ao utilizar o sistema pela primeira vez.		O sistema deverá indicar que não há veículos cadastrados no sistema	Ok
2. Teste de consistência – No teste 2 inserir informações inconsistentes.		O sistema deverá exibir uma mensagem indicando a inconsistência encontrada e retornar ao formulário.	ok
<input checked="" type="checkbox"/> Aprovado	Comentários		
<input type="checkbox"/> Reprovado			
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 04		Requisito testado: Gerir tipos de orientação	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações:		
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolhe a opção para listar os tipos de orientação.		O sistema deve exibir uma lista com todos os tipos de orientação cadastrados no sistema.	Ok
2. Escolher a opção para cadastro de novo tipo de orientação.		O sistema deve exibir o formulário de cadastro de tipo de orientação.	Ok
3. Escolher a opção de listar os tipos de orientação.		O sistema exibir a lista de tipos de orientação como o novo tipo de orientação cadastrado.	Ok
4. Escolher a opção para alterar o tipo de orientação previamente cadastrado.		O sistema deve exibir o formulário de alteração do tipo de orientação previamente cadastrado.	Ok
5. Escolher a opção de listar os tipos de orientação.		O sistema exibir a lista de tipos de orientação como o tipo de orientação que foi alterado.	Ok
6.			
7.			
Seqüência para verificação de exceções		Comportamento esperado do software	
1. Ao utilizar o sistema pela primeira vez.		O sistema deverá indicar que não há tipos de ocorrência cadastrados no sistema.	Ok
2.			
3.			
4.			
5.			
<input checked="" type="checkbox"/> <i>Aprovado</i> <input type="checkbox"/> <i>Reprovado</i>	Comentários		
Avaliadores: <i>Fabio, Gilson</i> Apresentador: <i>Daniel Konishi</i>			

Teste no. 05		Requisito testado: Gerir tipos de ocorrência	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações:		
Sequência para verificação do caso comum	Comportamento esperado do software	Testado	
1. Escolhe a opção para listar os tipos de ocorrência.	O sistema deve exibir uma lista com todos os tipos de ocorrência cadastrados no sistema.	Ok	
2. Escolher a opção para cadastro de novo tipo de ocorrência.	O sistema deve exibir o formulário de cadastro de tipo de ocorrência.	Ok	
3. Escolher a opção de listar os tipos de ocorrência.	O sistema exibir a lista de tipos de ocorrência como o novo tipo de ocorrência cadastrado.	Ok	
4. Escolher a opção de alterar o tipo de ocorrência do tipo previamente cadastrado.	O sistema deve exibir o formulário de alteração com o tipo de ocorrência previamente cadastrado.	Ok	
5. Escolher a opção de listar os tipos de ocorrência.	O sistema exibir a lista de tipos de ocorrência como o novo tipo de ocorrência cadastrado.	Ok	
6.			
7			
Sequência para verificação de exceções	Comportamento esperado do software		
1. Ao utilizar o sistema pela primeira vez.	O sistema deverá indicar que não há tipos de ocorrência cadastrados no sistema.	ok	
<input checked="" type="checkbox"/> <i>Aprovado</i> <input type="checkbox"/> <i>Reprovado</i>	Comentários		
Avaliadores Fabio, Gilson			
Apresentador Daniel Konishi			

Teste no. 06		Requisito testado: Gerir entregas	
Funcional: <input checked="" type="checkbox"/>	Observações:		
Não funcional: <input type="checkbox"/>			
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolhe a opção para listar as entregas em andamento.		O sistema deve exibir uma lista com todas as entregas que estão em andamento.	Ok
2. Escolher a opção para consultar uma entrega.		O sistema deve exibir as informações da entrega selecionada.	Ok
3. Escolher a opção de consultar um pedido.		O sistema deve exibir as informações do pedido selecionado.	Ok
4. Escolher a opção para enviar ocorrência.		O sistema deve exibir o formulário para envio de ocorrência.	Ok
5. Escolher a opção de listar os tipos de ocorrência.		O sistema exibir a lista de tipos de ocorrência como o novo tipo de ocorrência cadastrado.	Ok
6. Escolher a opção para visualizar orientação.		O sistema deverá exibir a nova orientação enviada.	Ok
7			
Seqüência para verificação de exceções		Comportamento esperado do software	
1.			
<input checked="" type="checkbox"/> Aprovado	Comentários		
<input type="checkbox"/> Reprovado			
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 07		Requisito testado: Envio de ocorrência	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações:		
Sequência para verificação do caso comum		Comportamento esperado do software	Testado
1. Escolhe a opção para enviar uma nova ocorrência.		O sistema deve exibir o formulário para envio de novo ocorrência.	Ok
2. Escolher a opção para lista as entrega e escolher a entrega para a qual enviamos essa nova ocorrência.		O sistema deve exibir a novo ocorrência enviada na lista de ocorrência daquela entrega.	Ok
3.			
4.			
5.			
6.			
7.			
Sequência para verificação de exceções		Comportamento esperado do software	
1.			
<input checked="" type="checkbox"/> Aprovado <input type="checkbox"/> Reprovado		Comentários	
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 08		Requisito testado: visualizar entrega (Agente externo)	
Funcional: <input checked="" type="checkbox"/>	Observações:		
Não funcional: <input type="checkbox"/>			
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. O agente externo entra no sistema e escolhe a opção de listar as entrega.		O sistema deve exibir uma lista de entrega do cliente ao qual o agente externo pertence.	Ok
2. Escolher a opção para consultar uma entrega.		O sistema deve exibir as informações detalhadas daquela entrega selecionada.	Ok
3. Escolher a opção para consultar um pedido.		O sistema deverá exibir as informações do pedido selecionado.	ok
4.			
5.			
6.			
7			
Seqüência para verificação de exceções		Comportamento esperado do software	
1.			
2.			
<input checked="" type="checkbox"/> Aprovado	Comentários		
<input type="checkbox"/> Reprovado			
Avaliadores Fabio, Gilson		Apresentador Daniel Konishi	

Teste no. 09		Requisito testado: Controle de acesso	
Funcional: <input checked="" type="checkbox"/> Não funcional: <input type="checkbox"/>	Observações: Fazer esse teste para todas as opções de permissão existente.		
Seqüência para verificação do caso comum		Comportamento esperado do software	Testado
1. Altera a permissão de acesso para esse funcionário.			Ok
2. Entra no sistema com o login desse funcionário.		O sistema deverá habilitar as opções para esse funcionário de acordo com a permissão de acesso dada.	ok
3.			
4.			
5.			
6.			
7			
Seqüência para verificação de exceções		Comportamento esperado do software	
1.			
2.			
<input checked="" type="checkbox"/> <i>Aprovado</i> <input type="checkbox"/> <i>Reprovado</i>	Comentários		
<div style="display: flex; justify-content: space-between;"> <div> Avaliadores Fabio, Gilson </div> <div> Apresentador Daniel Konishi </div> </div>			

ANEXO G – PADRÃO DE CODIFICAÇÃO

1. Definição de funções

Na declaração de uma função ou método de uma classe deverá ser utilizado o nome, em letra minúscula, da operação que essa função realiza (verbo). Caso seja um nome composto a primeira letra das palavras seguintes deverá começar com uma letra maiúscula.

Ex : orderList, showDetail, print.

2. Convenção de Nomes

Nesse projeto, os nomes dos arquivos do código fonte começam com a operação que é executada, seguido pelo objeto no qual a operação é executada.

Ex : searchClient, sendOrientation.

De forma semelhante, o nome das classes deverão começar como uma letra maiúscula, caso seja um nome composto a segunda parte do nome deverá começar com uma letra maiúscula.

Ex : Client, OrientationType.

Os objetos instanciados terão o mesmo nome da classe, porém, devendo começar com uma letra minúscula.

Ex: client, orientationType, externalAgent.