UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

GUSTAVO FERREIRA GOMES

ABORDAGEM DE UMA FERRAMENTA PARA COMUNICAÇÃO VIA WEBSERVICE COM ADMINISTRADORAS DE PAGAMENTO ELETRÔNICO DE FRETE QUE ATENDEM A RESOLUÇÃO DA ANTT Nº 3.658/11

CRICIÚMA

2013

GUSTAVO FERREIRA GOMES

ABORDAGEM DE UMA FERRAMENTA PARA COMUNICAÇÃO VIA WEBSERVICE COM ADMINISTRADORAS DE PAGAMENTO ELETRÔNICO DE FRETE QUE ATENDEM A RESOLUÇÃO DA ANTT Nº 3.658/11

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Gustavo Bisognin

CRICIÚMA

2013

GUSTAVO FERREIRA GOMES

ABORDAGEM DE UMA FERRAMENTA PARA COMUNICAÇÃO VIA WEBSERVICE COM ADMINISTRADORAS DE PAGAMENTO ELETRÔNICO DE FRETE QUE ATENDEM A RESOLUÇÃO DA ANTT Nº 3.658/11

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de Software

Criciúma, 25 de Novembro de 2013.

BANCA EXAMINADORA

Prof. Gustavo Bisognin – MSc. – (UNESC) – Orientador

Prof. Gilberto Vieira da Silva – Esp. (UNESC)

Prof. Fabrício Giordani – Esp. (UNESC)

RESUMO

A presente pesquisa refere-se ao levantamento de requisitos de software e modelagem lógica com abordagem de protótipo que visa unificar o desenvolvimento e forma de integração via Web Service, com administradoras de Pagamento Eletrônico de Frete que atendem a resolução 3.658/11 criada pela Agência Nacional de Transportes Terrestres (ANTT). Para a realização da pesquisa foi necessário o estudo das disciplinas de Engenharia de Software e Engenharia de Requisitos, bem como um estudo técnico no desenvolvimento de Web Services e uma busca avançada pelo conhecimento da legislação abordada. Após um levantamento de requisitos, e também de criar a modelagem lógica com prototipação, foi possível desenvolver um aplicativo padrão capaz de comunicar-se via Web Service com duas administradoras de pagamento eletrônico de frete, de forma simples e eficaz, atendendo desta forma a resolução vigente publicada pela ANTT.

Palavras-chave: Levantamento de requisitos. Modelagem lógica. Resolução 3.658/11. Web Services.

ABSTRACT

This research refers to software requirements elicitation and logical modeling that seeks to unify the development and integration via Web Service with administratores of electronic payment of freight to attending the brazilian law 3.658/11. Created by the National Agency Land Transportation (ANTT). To realize the research was necessary the study of the disciplines of Software Engineering and Engineering of Requirements such as a technical study on the development of Web Services and an specific knowledge about the actual legislation addressed. After a elicitation of requirements, and also create the logical modeling prototyping, it was possible to develop a standard app that is able to communicate via Web Service with two administratores of electronic payment of freight, this application is simple and effective, for attending the resolution of ANTT.

Keywords: Elicitation of Requirements. Logical Modeling. Resolution 3.658/11. Web Services.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ANTT	Agência Nacional de Transportes Terrestres
CIOT	Código Identificador da Operação de Transporte
CSP	*Constraint Satisfaction Problem*
EA	Enterprise Architech
ETC	Empresa de Transporte Rodoviário de Cargas
HTTP	*HyperText Transfer Protocol*
OASIS	*Organization for the Advancement of Structures Information Standards*
PEF	Pagamento Eletrônico de Frete
RNTRC	Registro Nacional de Transportadores Rodoviários de Cargas
SAML	*Security Assertion Markup Language*
SC	Santa Catarina
SOAP	*Simple Object Access Protocol*
TAC	Transportador Autônomo de Cargas
TCC	Trabalho de Conclusão de Curso
UDDI	*Universal Description, Discovery and Integration*
UML	*Unified Modeling Language*
URL	*Uniform Resource Locator*
XML	*eXtensible Markup Language*
W3C	*World Wide Web Consortium*
WS	*Web Service*
WSDL	*Web Service Description Language*

SUMÁRIO

1 INTRODUÇÃO

O procedimento de engenharia de requisitos é fundamental e também responsável por realizar a transformação de requisitos de usuário em requisitos de *software*. Este processo é a base para o ponto de partida de onde a qualidade é medida, desta forma, a falta de conformidade apresentada por este tipo de requisito demonstra ausência de qualidade.

As fábricas de *software* têm assumido um interesse explícito na engenharia de requisitos, pois possuem a necessidade de sempre entender o que se pretende construir antes de iniciar o que se desejou criar.

O custo elevado do esforço utilizado para o entendimento do problema é um excelente sinal para a elicitação de uma gerência efetiva, que seja controlada e adequada para a rápida compreensão dos requisitos de usuários de *software*.

É de grande importância que a empresa que deseja aumentar a qualidade de seus produtos esteja com um processo de engenharia de requisitos muito bem desenvolvido e aplicado nas equipes de desenvolvimento.

Decorrente da falta de tempo e entendimento dos requisitos presentes no *software*, muitos analistas acabam projetando alterações e criações de processos desunificados, totalmente isolados, e que se bem elicitados poderiam ser unificados e padronizados, amenizando e evitando custos e investimentos de tempos com mão de obra, visto que quando isso ocorre muitas outras alterações ficam pendentes, e o tempo se torna escasso para execução de rotinas necessárias no sistema.

O problema ganha potência no decorrer dos anos em que ocorre um aumento considerável do acumulo de trabalho (*backlog*), gerado decorrente do elevado número de alterações e atividades, fazendo com que muitas dessas atividades fiquem na fila de espera da empresa.

Motivado e diante do problema exposto, o presente estudo propõe, o levantamento de requisitos de *software* e modelagem lógica com abordagem de protótipo que unifique o desenvolvimento e forma da integração via Web Service (WS), de duas administradoras de Pagamento Eletrônico de Frete (PEF), respectivamente Apisul e Dbtrans, que atendem a resolução 3.658/11 criada pela Agência Nacional de Transportes Terrestres (ANTT).

1.1 OBJETIVO GERAL

Levantar os requisitos necessários e criar uma ferramenta protótipo para comunicação via web service com administradoras de pagamento eletrônico de frete que atendem a resolução da ANTT nº 3.658/11.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos do projeto que se pretende atingir com a proposta de trabalho são os seguintes:

a) realizar levantamento bibliográfico;
b) pesquisar sobre o que a ANTT exige na resolução 3.658/11;
c) estudar modelos de layout das administradoras PEF;
d) realizar levantamento e definir requisitos de *software*;
e) criar modelagem lógica que será utilizada para integração com as administradoras PEF;
f) desenvolver o protótipo;
g) realizar testes do protótipo, realizando os testes de integração via web service;
h) avaliar se a comunicação com ambas as administradoras PEF realizou-se com sucesso.

1.3 JUSTIFICATIVA

O tema abordado reveste-se de importância nos assuntos relacionados à engenharia de *software*.

Conforme mencionado por Pressman (2006, p. 17), “a engenharia de *software* é a criação e a utilização de sólidos princípios de engenharia a fim de obter *softwares* econômicos que sejam confiáveis e que trabalhem eficientemente em máquinas reais”. Neste sentido a engenharia também deve conciliar a qualidade ao prazo estimado de entrega, atendendo a real necessidade do cliente.

A necessidade da abordagem do assunto do presente trabalho surgiu após a publicação de uma resolução criada pela ANTT.

Segundo a empresa Dbtrans (2012), com a proibição da carta-frete

(sistemática ineficiente de pagamento de frete) e a regulamentação das administradoras de pagamento do transporte de carga, em abril de 2011 através da resolução 3.658/11 da ANTT, os caminhoneiros e transportadoras estão unindo esforços para adaptarem-se as novas obrigações exigidas na resolução. As regras elaboradas pela Agência Nacional de Transportes Terrestres criaram a figura das administradoras para fazer o vinculo entre o contratante do frete (transportadora), o caminhoneiro (terceirizado contratado) e o órgão regulador (ANTT).

Devido à existência de particularidades em cada administradora de pagamento eletrônico de frete a uma grande preocupação com o tempo gasto e com a qualidade no desenvolvimento e atualizações de *softwares* comerciais que integram com estas administradoras de PEF.

Analisando a definição do problema, observa-se certo grau de dificuldade do desenvolvedor em cumprir os prazos devido às características destas administradoras, e como trata-se de legislação é imprescindível que os prazos estipulados pela ANTT na resolução sejam rigorosamente respeitados.

Com isso, visando à diminuição efetiva do tempo despendido para integração com mais de uma administradora de pagamento eletrônico de frete, será realizado o levantamento de requisitos, com criação de modelagem lógica e também será implementado uma ferramenta protótipo para comunicação via Web Service com estas administradoras que atendem a resolução nº 3.658/11 da Agência Nacional de Transportes Terrestres.

1.4 ESTRUTURA DO TRABALHO

O corrente trabalho possui uma estrutura de sete capítulos dispostos em forma organizada.

No capítulo que inicia ao trabalho, é mostrada a introdução, os objetivos gerais e específicos, a justificativa e a presente descrição da estrutura do projeto proposto. O segundo capítulo tem como foco mostrar conceitos relacionados à engenharia de *software*, suas características, seus processos, seus modelos, onde de forma geral será abordado no referente projeto. O capítulo de número três aborda a prática da engenharia de requisitos e a modelagem de sistemas da informação, na qual a modelagem lógica de dados a compõe. No quarto capítulo é descrito sobre Web Service, bem como sua padronização, arquitetura e fluxo de informações

percorridas no serviço da web. O quinto capítulo trata-se de um assunto legislativo e técnico, que são as exigências da ANTT, conforme a resolução abordada neste trabalho. No capítulo seis são abordados de forma sintética os trabalhos correlatos. No sétimo e último capítulo é mencionado o trabalho juntamente com as metodologias, recursos utilizados. Por fim, o trabalho é finalizado com elementos pós textuais.

2 ENGENHARIA DE *SOFTWARE*

O capítulo corrente contextualizará a evolução e criação de *software*, e também mencionará algumas das dificuldades que surgem no processo de desenvolvimento de *software* bem como apresentar o quão importante foi à criação da área de engenharia de *software*, onde a partir disso os *softwares* passaram a ter maior reconhecimento e abrangência de mercado e, de forma conseqüente diminuiu a ocorrência de inconsistências na etapa de utilização dos *softwares*.

2.1 CONCEITUAÇÃO DE SOFTWARE

Conforme Pressman (2006), os *softwares* possuem um duplo papel: em momento é o produto, em momentos é o que veicula o produto.

O *software*, enquanto produto assume um potencial de computação que pode ser encontrado de forma freqüente em aparelhos móveis ou, até mesmo, em computadores de grande porte. O *software* é o que transforma a informação.

Quando visualizado como veículo de entrega do produto, o *software* age como uma base de controle do computador, possibilitando a comunicação da informação (rede) e, também, a criação e o controle de ambientes desenvolvedores de *software* e ferramentas.

Um *software* deve ser projetado e implementado de modo que possa ser reaproveitado no desenvolvimento de outros, como ocorre na reutilização das interfaces gráficas que podem ser reaproveitadas pelo engenheiro de *software* na criação de novas aplicações.

Segundo Pressman (2006) atualmente os engenheiros de *software* estão encontrando dificuldades em sete categorias de *softwares* de computadores, sendo as seguintes:

a) sistemas: são os programas responsáveis por dar apoio a outros programas;
b) aplicação: são as aplicações responsáveis pela especificidade de determinado negócio;
c) científico e de engenharia: são aplicações modernas utilizadas na área científica da engenharia que estão se distanciando dos algoritmos numéricos convencionais, isto é, estão adotando um novo formato de

desenvolvimento;

d) embutidos: acoplados dentro de um sistema ou produto e são utilizados para implementar e controlar características e funcionalidades específicas que o próprio sistema ou produto não atende;

e) linhas de produtos: são os programas ou aplicações projetados para fornecer capacidade específica para clientes diferentes. Um exemplo deste tipo *software* são os sistemas de sincronização bancária;

f) aplicações da web: *softwares* diversos que são acessados via web e que englobam uma área de amplas aplicações;

g) inteligência artificial: são os programas que utilizam algoritmos não numéricos, e que são responsáveis por resolver programas de grande complexidade que não são entendidos facilmente por análise direta ou computadores.

Um projeto de *software* é iniciado em razão da necessidade de determinado negócio, sendo que a necessidade pode ser proveniente de várias situações, tais como: correção de defeito de aplicações existentes, adaptação de um sistema em funcionamento para mudar o ambiente de negócio, extensão das funções e características de uma aplicação existente ou desenvolvimento de um produto novo. Em diversas situações, as necessidades de um novo projeto de *software* são expressas informalmente, partindo de uma simples conversa (PRESSMAN, 2006).

2.2 HISTÓRICO DA ENGENHARIA DE SOFTWARE

De acordo com Sommerville (2003, p. 6), “engenharia de *software* é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de *software*, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema”.

Conforme o autor, o conceito de engenharia de *software* surgiu durante a conferência da “Crise de *Software*”, que ocorreu em 1968 e foi resultante da introdução direta de novos hardwares de computadores baseados em circuitos integrados, desta forma, os *softwares* que vinham sendo desenvolvidos não estavam de acordo com os recursos fornecidos pelos novos hardwares.

Segundo Pressman (2006) a prática da engenharia de *software* foi adotada por decorrência da preocupação existente na indústria de *software*, pela forma em

que os *softwares* são desenvolvidos.

A engenharia de *software*, além de tornar ampla as garantias relacionadas à qualidade do produto, é a área de conhecimento específico que dá auxílio ao desenho de soluções de *software* que é utilizado para dar resposta a complexidade crescente e a diversidade dos sistemas, assim como as alterações nos requisitos durante o seu processo de desenvolvimento.

Considerando a engenharia de *software* uma disciplina que ainda não atingiu a maturidade de outros ramos da engenharia, é muito importante a existência de processos que garantam qualidade e controle do desempenho de atividades nesta área. Desta forma, o papel dos recursos humanos nesse processo é um fator decisivo para o sucesso dos projetos. Segundo Boot (1995), existem características desejáveis em sua gestão:

a) planejamento estratégico para longo prazo, assegurando-se assim uma plataforma de orientação para todos os intervenientes;
b) empenho na organização, passando-se do mero cumprimento de regras e prazos para uma atitude de empenho e auto-motivação;
c) autogestão pelos intervenientes, cabendo, à gestão, a delegação de tarefas e a anulação de um controle sufocante;
d) perspectiva unitária, garantindo-se que os esforços individuais se conjuguem para um melhor desempenho em equipe e premiando-se o mérito individual;
e) promoção da existência de intervenientes flexíveis.

De forma resumida, pode-se afirmar que não existem processos de desenvolvimento de *software* mais ou menos corretos ou meio corretos, o que existe são processos de desenvolvimento mais ou menos adequados à complexidade do projeto solicitado, e ao tipo de equipe que o desenvolve e também ao tipo de utilização que se pretende para o produto de *software* final.

2.3 MODELOS DE DESENVOLVIMENTO DE SOFTWARE

Um processo de desenvolvimento de *software* trata-se um conjunto de atividades e resultados associados que auxiliam na produção de *software* (SOMMERVILLE, 2003).

Dentre as várias atividades que estão associadas, pode-se citar a análise

de requisitos e a codificação. Com o resultado do processo de *software* chega-se a um produto que reflete a forma como o processo foi conduzido. Verifica-se que diferentes organizações utilizam-se de processos diferentes para produção de um mesmo produto e alguns processos são mais adequados do que outros dependendo da aplicação. Embora existam vários processos para o desenvolvimento de *software*, existem atividades fundamentais comuns a todos eles. O quadro 1 demonstra tais atividades.

Quadro 1 – Atividade para o Desenvolvimento de *Software*.

Atividades para o desenvolvimento do *software*	
Especificação de *Software*	Definição das funcionalidades (requisitos) e das restrições do *software*.
Projeto de implementação de *Software*	O *software* é produzido de acordo com as especificações.
Validação de *Software*	O *software* é validado para garantir as funcionalidades.
Evolução de *Software*	O *software* precisa evoluir para continuar sendo útil ao cliente.

Fonte: Adaptado de Sommerville (2003).

A especificação de *software* refere-se à fase em que é definido com o solicitante as características do *software*. Em uma segunda fase no projeto e implementação de *software*, propõe-se modelos de diagramas, que posteriormente são implementados em alguma linguagem de programação. A fase seguinte relaciona-se a validação do *software*, que se refere à garantia de que todas as funcionalidades especificadas foram implementadas corretamente. A última atividade trata-se da evolução do *software*, o que assegura o solicitante de que o *software* continuará com utilidade. Ainda referindo-se a modelos de processos, encontram-se vários processos de *software* definidos na literatura da Engenharia de *Software*. Dentre os vários processos, existem as metodologias tradicionais, que são orientadas a documentação, e as metodologias ágeis, que procuram desenvolver *software* com o mínimo de documentação possível.

Para Royce (1970 apud SOARES, 2004, p. 2),

> as metodologias tradicionais são também chamadas de pesadas ou orientadas a documentação. Essas metodologias surgiram em um contexto de desenvolvimento de *software* muito diferente do atual, baseado apenas em um *mainframe* e terminais burros.

A utilização desta metodologia, tornava o *software* planejado e documentado antes de ser implementado. Dentre as principais metodologias

tradicionais, utiliza-se até os dias de hoje o Modelo Clássico.

Segundo Soares (2004, p. 5), nas “metodologias ágeis o enfoque nas pessoas e não em processos ou algoritmos”. Com isso, gasta-se menos tempo com documentação e mais com a implementação, tornando-se adaptativas ao invés de serem preditivas.

Soares (2004, p. 36) ainda observa, que normalmente algumas organizações criam seu próprio processo ou adaptam à sua realidade. Os desenvolvedores de *software* implementam seu sistema sem usar nenhum processo. Isso ocorre porque o processo habitual não se adequou à necessidade da organização. Em particular, as organizações pequenas e médias não possuem recursos suficientes para adotar o uso de processos pesados. Por esta razão, muitas organizações não utilizam nenhum processo. O resultado desta falta de sistematização na produção de *software* é a baixa qualidade do produto final, além de dificultar a entrega nos prazos e custos predefinidos e inviabilizar a futura evolução. Um modelo de desenvolvimento corresponde a uma representação abstrata do processo que por fim direcionará até o alcance do objetivo, que se trata de um produto de baixo custo e alta qualidade. Apresentam-se na seqüência, alguns modelos conhecidos e utilizados no desenvolvimento de *softwares*.

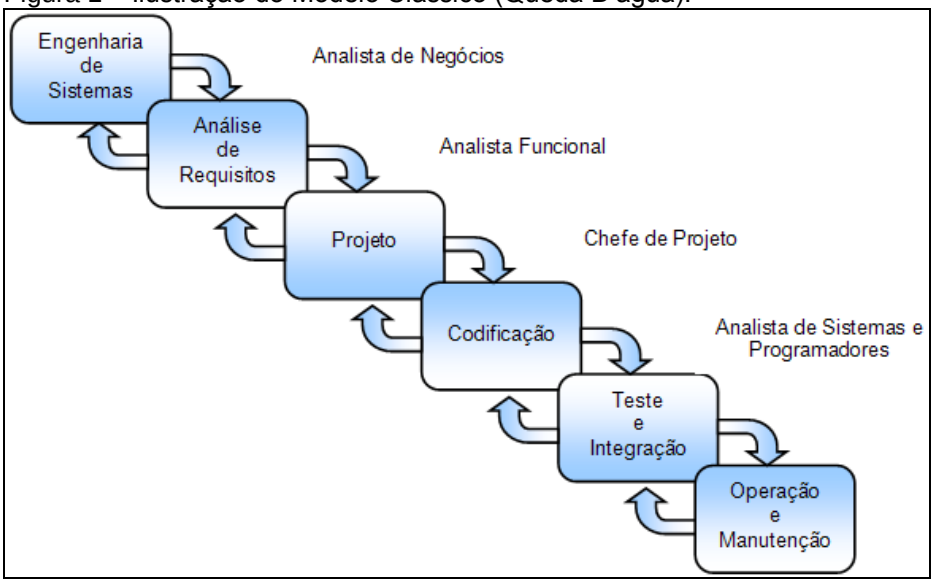
2.3.1 Modelo Clássico

O modelo clássico também chamado de Modelo em Cascata compreende em uma sistemática, seqüencial ao desenvolvimento do *software*.

O modelo de cascata descreve um método de desenvolvimento linear e seqüencial permitindo um controle departamental e gerencial, seguindo uma ordem estrita, sem qualquer sobreposição ou passos iterativos (PRESSMAN, 2006).

É representado conforme figura 1, o ciclo mais simples de desenvolvimento de *software*, estabelecendo o qual obedece a uma ordenação linear no que diz respeito à realização das diferentes etapas.

Figura 1 – Ilustração do Modelo Clássico (Queda D'água).



Fonte: Adaptado de Pressman (2006).

No Ciclo de Vida Clássico utilizam-se conceitos de Engenharia de *Software*, a qual prevê atividade de verificação, validação e de controle de qualidade (PARREIRA JUNIOR, 2011). Trata-se de um paradigma que utiliza um método sistemático e seqüencial em que o resultado de uma fase se constitui na entrada de outra fase, abrangendo as seguintes atividades ilustradas na figura 1.

Quadro 2 – Atividades do Ciclo de Vida Clássico.

Atividades do Ciclo de Vida Clássico	
Engenharia de Sistemas	Quanto mais dados forem coletados em nível de sistema, menor será a probabilidade de haver "bugs" no sistema.
Análise de Requisitos	Saber o que o cliente quer que o *software* tenha, com relação a recursos, sendo documentados e revistos antes de começar a execução do projeto.
Projeto	Envolvem 4 atributos distintos do programa: estrutura de dados, arquitetura de *software*, detalhes procedimentais e caracterização de interface.
Codificação	O projeto deve ser transformado em programa, usando uma linguagem de programação, isto é, traduzido numa linguagem de máquina.
Teste e Integração	Testa-se os programas. A Integração consiste das junções das unidades e programas desenvolvidos concordando com o projeto.
Operação e Manutenção	O sistema é instalado e colocado em uso. A manutenção ocorre com a correção de erros não detectados, e na adição de novas características.

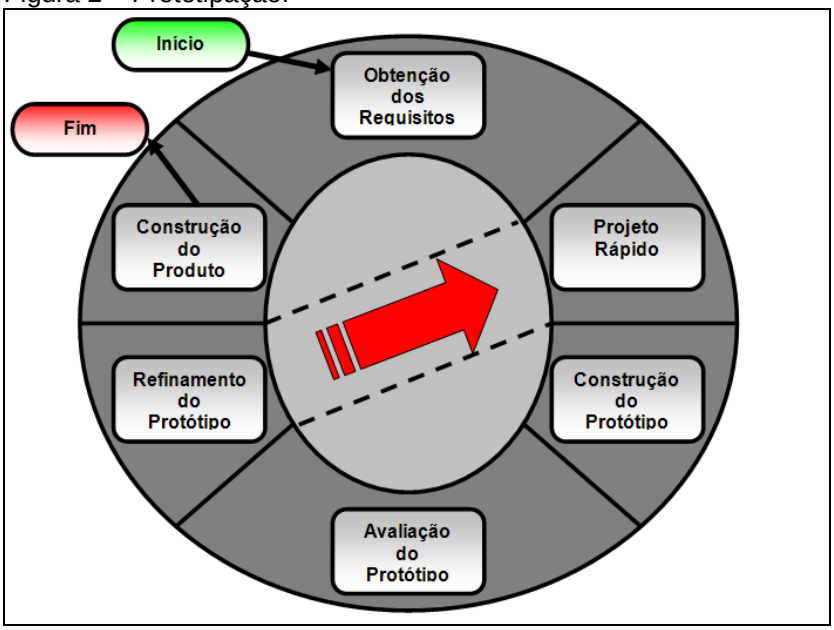
Fonte: Adaptado de Parreira Junior (2011).

O ciclo de vida clássico refere-se ao paradigma mais antigo e o mais amplamente usado em Engenharia de *Software*, no entanto deve-se atentar aos detalhes, pois corre-se riscos de não se atingir o objetivo final, tornando-se um produto de alto custo para a realização de correções.

2.3.2 Prototipação

Este modelo de desenvolvimento, que se baseia em um protótipo relaciona-se a uma versão preliminar do sistema apresentado ao usuário, onde ele fornece informações ao desenvolvedor, para que sejam realizadas adaptações e implementações, durante o projeto e desenvolvimento. Segundo Mazzola (2010, p. 110), “o objetivo da prototipação é um modelo de processo de desenvolvimento que busca contornar limitações existentes no modelo cascata, eliminando a política de congelamento dos requisitos antes do projeto ou codificação”.

Figura 2 – Prototipação.



Fonte: Adaptado de Pressman (2006)

De acordo com Oliveira (2009, p. 6), no modelo prototipagem (pura), “o desenvolvedor interage diretamente com o usuário, escutando seus pedidos e desenvolvendo, imediatamente, um protótipo do produto desejado”. Com este modelo, o usuário utiliza o protótipo e fornece ao desenvolvedor outras novas informações que o levam a gerar atualizações, adaptações e implementações no *software*, durante o de projeto e desenvolvimento. Desta forma, observa-se que a prototipagem trata-se de uma técnica valiosa no levantamento de requisitos, tornando-se mais reais, facilitando o entendimento. Com isso, ao colocar o usuário na frente de parte ou imitação do sistema, a prototipagem estimula os usuários a pensar e a estabelecer um diálogo sobre os requisitos (FALBO, 2011). Este modelo permite que diferentes

tipos de protótipos possam ser desenvolvidos, conforme apresentado no quadro 3.

Quadro 3 – Tipos de protótipos.

Tipos de Protótipos Quanto:		
Arquitetura de Sistema	**Uso do Protótipo**	**Funcionalidades**
Protótipo não-operacional ou de interface	Protótipo descartável	Protótipo de características selecionadas
Protótipo operacional	Protótipo evolutivo	Protótipo completo

Fonte: Adaptado de Falbo (2011).

Com relação às camadas da arquitetura de sistema que são efetivamente implementadas, o protótipo relaciona-se ao tipo operacional e não operacional.

No protótipo não-operacional ou de interface, apenas a camada de interface com o usuário é implementada, desta forma, o sistema não faz nenhum processamento propriamente dito, já o protótipo operacional funciona como o sistema real deveria funcionar e implementar todas as camadas da arquitetura do sistema (FALBO, 2011).

Referindo-se ao uso futuro do protótipo, como base para o sistema real ou não, um protótipo pode ser descartável, exploratório e não se pretende utilizá-lo como uma parte real do sistema a ser fornecido ou evolutivo, desenvolvido para se aprender mais sobre o problema e se ter a base de uma parte ou de todo o *software* a ser fornecido.

De acordo com o conjunto de funcionalidades fornecidas pelo protótipo, ele pode ser de características selecionadas ou um protótipo completo. No protótipo de características selecionadas, apenas uma porção do sistema é implementada. Já o protótipo completo, apresenta todas as características do que se imagina ser o sistema real (FALBO, 2011).

Compreende-se as particularidades das diferentes classificações de protótipos e que, no entanto podem ser combinadas, mas com seus devidos cuidados para que não se torne deficiente.

2.3.3 Modelo Espiral

O modelo espiral foi sugerido posteriormente ao modelo cascata em 1988 por Barry Boehm, com a intenção de abranger as melhores características, no ciclo de vida clássico como da prototipação, acrescentando ao mesmo tempo, um novo

elemento: a análise de riscos que falta a esses paradigmas. O quadro 4 descreve as quatro importantes atividades do modelo espiral, representadas por quatro quadrantes.

Quadro 4 – Atividades do Modelo Espiral.

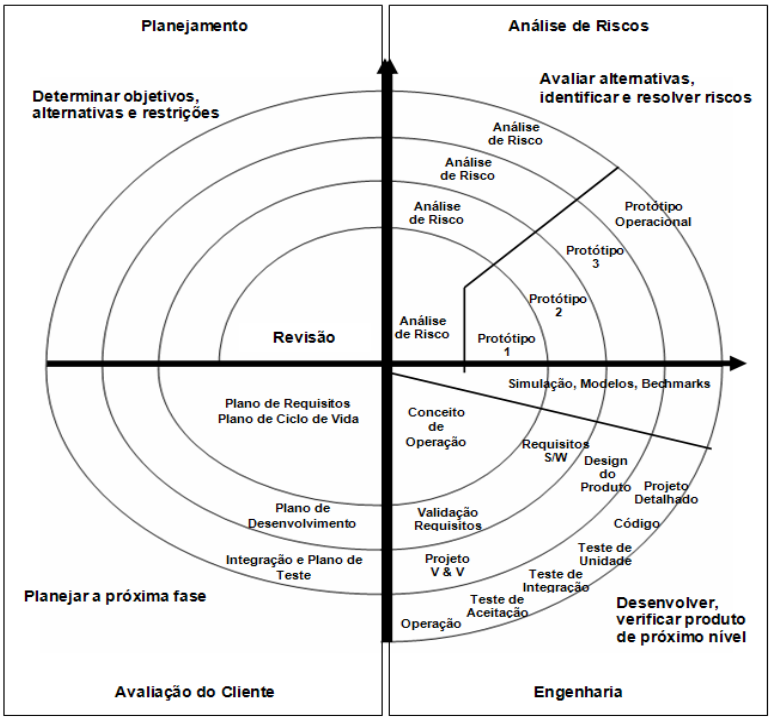
Modelo Espiral – Atividades	
Planejamento	Determinação dos objetivos, alternativas e restrições.
Análise de riscos	Análise de alternativas e identificação/resolução de riscos.
Engenharia	Desenvolvimento do produto no “nível seguinte”.
Atualização feita pelo cliente	Avaliação dos resultados da engenharia.

Fonte: Adaptado de Lessa e Lessa Junior (2010).

No modelo espiral mostra-se que as diferentes atividades são repetidas até uma decisão ser tomada e o documento de especificação de requisitos ser aceito. Compreende-se com este modelo, como sendo de uma abordagem “evolucionária” à engenharia de *software*, capacitando o desenvolvedor e o cliente a entender e reagir aos riscos em cada fase evolutiva.

O modelo espiral exige uma consideração direta dos riscos técnicos em todas as etapas do projeto e, se adequadamente aplicado, deve reduzir os riscos antes que eles se tornem problemáticos (PRESSMAN, 2006).

Figura 3 – O Modelo Espiral.



Fonte: Adaptado de Sommerville (2003).

Na figura 3, verifica-se uma avaliação maior dos riscos de desenvolvimento de programa, bem como o protótipo obtido no passo corrente já resolve boa parte dos riscos ligados a desempenho, desta forma percebe-se a evolução segundo o modelo Queda D'água.

Verifica-se também que o elemento que conduz este processo trata-se da consideração sobre os riscos, o que permite a adequação a qualquer política de desenvolvimento. A continuidade do processo de desenvolvimento é definida como função dos riscos remanescentes, como por exemplo, a decisão se os riscos relacionados ao desempenho ou à interface são mais importantes do que aqueles relacionados ao desenvolvimento do programa (MAZZOLA, 2010).

Considera-se uma característica importante, o encerramento de cada ciclo por uma atividade de revisão, onde todos os produtos do ciclo são avaliados, incluindo o plano para o próximo ciclo. No quadro 5 menciona-se as atividades do ciclo do modelo espiral, e sucintamente a rotina de cada etapa do ciclo.

Quadro 5 – Atividades do Ciclo do Modelo Espiral.

Atividades do Ciclo do Modelo Espiral		
Fases do Ciclo	**Atividades**	**Rotina**
	Elaboração	Elaborar objetivos, restrições e alternativas para entidades de *software*.
	Avaliação	Avaliar alternativas, restrições, e identificar as principais fontes de riscos.
	Desenvolvimento	Elaborar a definição das entidades de *software* em um projeto.
	Próxima Fase	Planejar o próximo ciclo. Abortar um projeto se apresentar riscos.

Fonte: Adaptado de Lessa e Lessa Junior (2010).

Observa-se que o Modelo de Desenvolvimento em Espiral, surgiu com o objetivo de resolver os problemas dos projetos que seguem o fluxo que o modelo Cascata propõe. Entretanto, no início do projeto torna-se difícil para o cliente especificar os requisitos. Com isso, espera-se que os requisitos básicos estejam bem entendidos, e necessita-se que os detalhes sejam bem definidos. Em conseqüência com o passar do tempo, o produto evoluirá em versões mais completas.

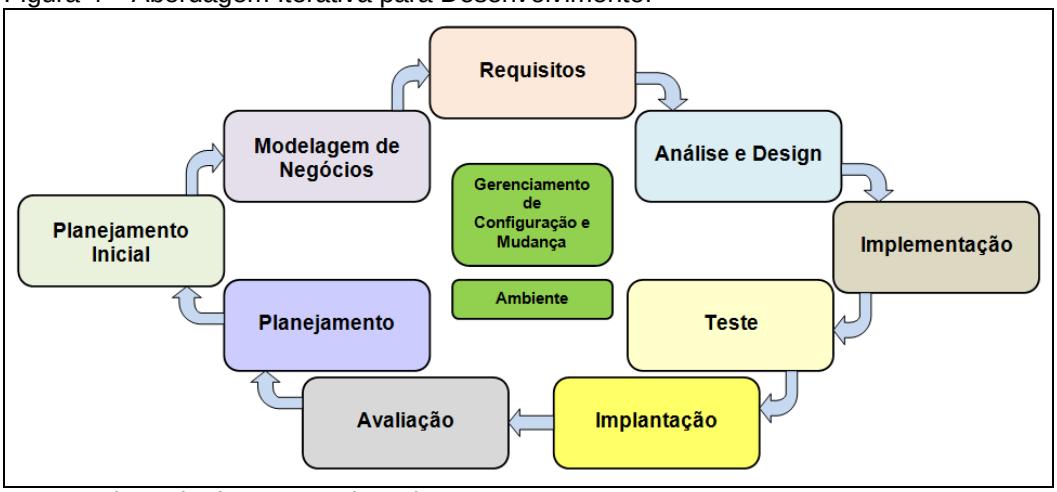
2.3.4 Desenvolvimento Iterativo e Incremental

É necessário que um processo de desenvolvimento de *software* seja iterativo, ou seja, ter várias iterações no tempo. Como também é necessário que seja incremental, ou seja, gerar novas versões incrementadas a cada atualização. Para

Silva, Souza e Dantas (2006, p. 18), “o modelo iterativo e incremental é uma alternativa para solução de problemas enfrentados no modelo cascata”.

Uma alternativa à abordagem em cascata é o processo iterativo e incremental, conforme mostrado na figura abaixo (figura 4).

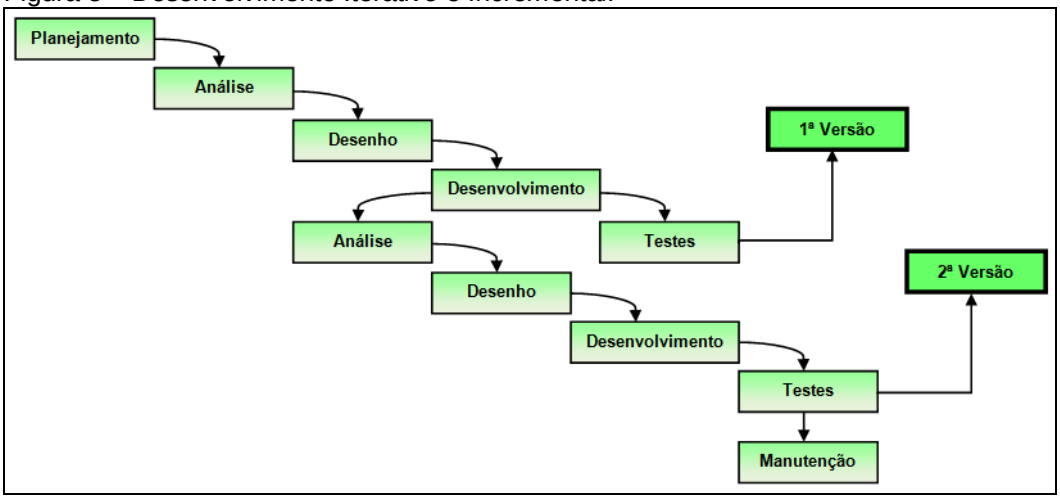
Figura 4 – Abordagem Iterativa para Desenvolvimento.



Fonte: Adaptado de Bezerra (2003).

De acordo com Bezerra (2003, p. 1), “um processo de desenvolvimento segundo essa abordagem divide o desenvolvimento de um produto de *software* em ciclos”. Neste modelo de desenvolvimento, considera-se o desenvolvimento do *software* em ciclos iterativos, onde uma parcela dos requisitos passa por todas as etapas de desenvolvimento, semelhante ao modelo de cascata.

Figura 5 – Desenvolvimento Iterativo e Incremental.



Fonte: Adaptado de Bezerra (2003).

Conforme a figura 5, visualiza-se que a cada ciclo de iteração têm-se uma

nova versão do *software*, onde deve ser incrementada a cada novo ciclo.

Segundo Silva, Souza e Dantas (2006, p. 18), “ao final de cada ciclo de iteração pode-se ter uma versão do *software*”. Neste sentido, a expectativa e a ansiedade do usuário em relação ao *software* diminuem. O modelo em questão possibilita desenvolvimento do *software* em módulos de acordo com as necessidades e características do projeto. Não é descartado o planejamento formal neste modelo, as iterações e idéias permitem uma compreensão e refinamento a cada etapa, promovendo uma evolução gradativa.

A conceituação incremental e iterativa somente torna-se possível existindo um mecanismo que divida os requisitos do sistema em partes, sendo que cada parte é alocada a um ciclo de desenvolvimento. A alocação realiza-se em função do grau de importância atribuído a cada requisito, entretanto, este modelo como qualquer outro apresenta suas vantagens e desvantagens conforme descritas no quadro 6.

Quadro 6 – Vantagens e Desvantagens do Modelo Iterativo e Incremental.

Modelo Iterativo e Incremental	
Vantagens	**Desvantagens**
Redução dos riscos envolvendo custos a um único incremento. - Se os desenvolvedores precisarem repetir a iteração, a organização perde somente o esforço mal direcionado de uma iteração, não o valor de um produto inteiro.	Dificuldade de gerenciamento. - Isso ocorre porque as fases do ciclo podem estar ocorrendo de forma simultânea.
Redução do risco de lançar o projeto no mercado fora da data planejada. - Identificando os riscos numa fase inicial o esforço despendido para gerenciá-los ocorre cedo, quando as pessoas estão sob menos pressão do que numa fase final de projeto.	O usuário pode se entusiasmar excessivamente com a primeira versão do sistema e pensar que tal versão já corresponde ao sistema como um todo.
Aceleração do tempo de desenvolvimento do projeto como um todo. - Porque os desenvolvedores trabalham de maneira mais eficiente quando buscam resultados de escopo pequeno e claro.	Como todo modelo esta sujeito a riscos de projeto. - O projeto pode não satisfazer aos requisitos do usuário, a verba do projeto pode acabar. - O sistema de *software* pode ser entregue ao usuário tarde demais.
Reconhecimento de uma realidade freqüentemente ignorada. - As necessidades dos usuários e os requisitos correspondentes não podem ser totalmente definidos no início do processo. Eles são tipicamente refinados em sucessivas iterações. Este modelo de operação facilita a adaptação a mudanças de requisitos.	

Fonte: Adaptado de Bezerra (2003).

Todas as vantagens e desvantagens descritas se relacionam com a implantação de qualquer modelo, pois os entraves são normais, claro que de acordo com às particularidades de desenvolvimento de cada um. As expectativas e frustrações ocorridas com o *software*, bem como os riscos no desenvolvimento, fica a cargo dos responsáveis pelo projeto de gerenciá-los, para que não gere desconforto e custos exagerados como também atraso no prazo de entrega do projeto.

2.3.5 Metodologia Ágil

A constante evolução no desenvolvimento de *softwares* tornou-se uma referência quando um grupo de especialistas estabeleceu alguns quesitos. Desde o ocorrido, as metodologias ágeis são apontadas como uma alternativa às abordagens mais tradicionais no desenvolvimento de *software*.

Banki e Tanaka (2008, p. 22) afirmam que “as metodologias ágeis, como a Extreme Programming (XP) e o Scrum, entre outras, têm despertado atenção crescente do mercado”.

Entre os conhecidos métodos ágeis, destaca-se a Programação Extrema (XP), criada a partir de inúmeras práticas de sucesso adotadas na indústria e formalizada no ano de 1999 por Kent Beck, com a publicação do livro “*Extreme Programming Explained: Embrace Change*”. A Programação Extrema dispõe um conjunto de valores, princípios e práticas, que possuem como visão garantir o sucesso no desenvolvimento de *software*, em face a requisitos vagos e que mudam constantemente (SATO, 2007).

Este movimento, que é baseado no ciclo de desenvolvimento incremental e iterativo, possui como foco a colaboração do cliente, no valor dos indivíduos e na adaptação às mudanças. Então, demonstram-se com isso, grandes ganhos de produtividade nos diversos tipos de projetos de desenvolvimento de *software*, eliminando-se documentação em excesso e burocracia, proporcionando interatividade e qualidade no desenvolvimento de *software*.

No ano de 2003, em um artigo publicado, Martin Fowler, cita algumas metodologias que estão representadas no quadro 7.

Quadro 7 – Metodologias Ágeis.

Metodologias Ágeis citadas por Martin Fowler	
Metodologias	XP (*Extreme Programming*)
	Scrum
	Crystal
	Context Driven Testing
	Lean Development
	(Rational) Unified Process
	Agile Modeling
	DSDM *(Dynamic Systems Development Method)*
	FDD *(Feature Driven Development)*
	Pragmatic Programming
	Adaptive Software Development

Fonte: Adaptado de Fowler (2003).

Para Ludvig e Reinert (2007, p. 3), “o termo Metodologia Ágil, popularizou-se em fevereiro de 2001, quando um grupo de 17 especialistas criou a Aliança Ágil e estabeleceram o Manifesto Ágil para o desenvolvimento de *software*”.

Quadro 8 – Manifesto Ágil.

Valores do Manifesto Ágil	
“Manifesto Ágil” (Agile Manifesto)	Indivíduos e interações são mais importantes que processos e Ferramentas.
	Software funcionando é mais importante do que documentação completa e detalhada.
	Colaboração com o cliente é mais importante do que negociação de contratos.
	Adaptação a mudanças é mais importante do que seguir o plano inicial.

Fonte: Adaptado Ludvig e Reinert (2007).

Para dar suporte na compreensão do modelo de desenvolvimento ágil, os membros da Aliança Ágil refinaram as filosofias contidas em seu manifesto.

O quadro 9 relaciona-se aos doze princípios, nos quais os métodos ágeis de desenvolvimento de *software* devem se adequar.

Quadro 9 – Doze Princípios do Manifesto Ágil.

Doze Princípios do Manifesto Ágil		
Princípios	1	A prioridade é satisfazer ao cliente através de entregas de software de valor contínuas e freqüentes.
	2	Entregar *softwares* em funcionamento com frequência de algumas semanas ou meses, sempre na menor escala de tempo.
	3	Ter o software funcionando é a melhor medida de progresso.
	4	Receber bem as mudanças de requisitos, mesmo em uma fase avançada, dando aos clientes vantagens competitivas.
	5	As equipes de negócio e de desenvolvimento devem trabalhar juntas diariamente durante todo o projeto.
	6	Manter uma equipe motivada fornecendo ambiente, apoio e confiança necessário para a realização do trabalho.
	7	A maneira mais eficiente da informação circular dentro da equipe é através de uma conversa face a face.
	8	As melhores arquiteturas, requisitos e projetos provêm de equipes organizadas.
	9	Atenção contínua a excelência técnica e um bom projeto aumentam a agilidade.
	10	Processos ágeis promovem o desenvolvimento sustentável. Todos envolvidos devem ser capazes de manter um ritmo de desenvolvimento constante.
	11	Simplicidade é essencial.
	12	Em intervalos regulares, a equipe deve refletir sobre como se tornarem mais eficazes e então se ajustar e adaptar seu comportamento.

Fonte: Adaptado por Ludvig e Reinert (2007).

Para a definição da metodologia mais adequada para o desenvolvimento de *software* em uma organização, levam-se em consideração os fatores envolvidos. De outra forma, trata-se de um fator principal para o sucesso da organização. Embora um processo adequado não possa garantir o sucesso total de um projeto, certamente a adoção de um processo não adequado pode comprometê-lo.

3 ENGENHARIA DE REQUISITOS

Além de definir requisito de *software*, neste capítulo serão apresentadas informações importantes acerca do assunto, tais como: evolução e no que baseia-se a sua criação, a necessidade de obtê-lo formalizada e detalhadamente e também formas de a representar dentro de um projeto de *software* bem estruturado e definido. Os requisitos de *software* são todas as necessidades funcionais e não funcionais que o *software* terá que apresentar para atender a necessidade do solicitante considerando a abrangência do negócio.

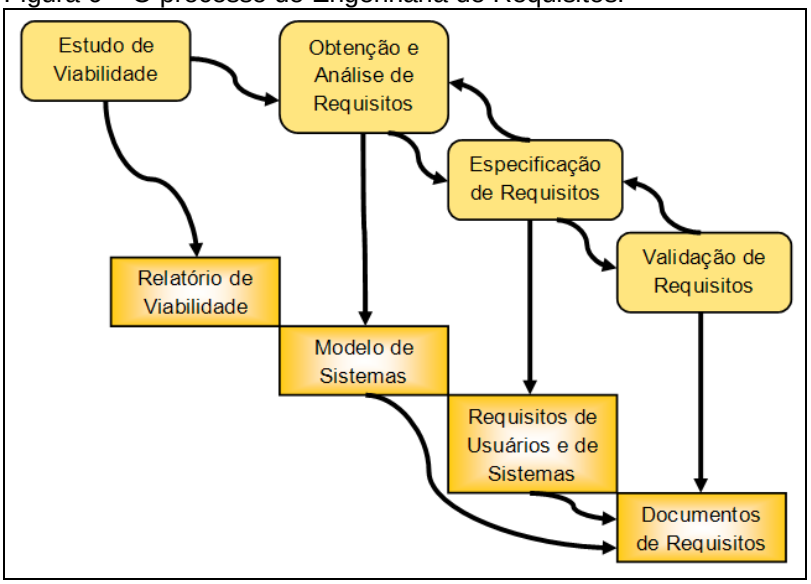
Pressman (2006) afirma que compreender os requisitos de um problema é uma das atividades mais difíceis de um engenheiro de *software*, mas levando em consideração o pensar e agir, o entendimento torna-se mais fácil. Este mesmo autor também afirma que a construção de um projeto de *software* é desafiadora, criativa e prazerosa, tanto que em muitos casos mesmo que os técnicos não possuam todas as informações necessárias, os mesmos ficam ansiosos para iniciar o desenvolvimento do *software*.

A engenharia de requisitos, bem como as demais atividades do processo, precisa ser adaptada às necessidades do processo, da tarefa, do projeto, das pessoas envolvidas e, principalmente do produto. Pois estas atividade são de suma importância o desenvolvimento do projeto.

Esta engenharia compõe uma importante parte da engenharia de *software*, onde há grande investimento de empresas no aprimoramento técnico, e também o processo onde se aponta os serviços que o sistema deve disponibilizar e as restrições que deve limitar-se, firmando uma estrutura sólida para o projeto e construção. Com a ausência da engenharia, a conclusão do *software* tem a possibilidade de ser insatisfatória para as necessidades dos solicitantes do *software*.

De acordo com Falbo (2011, p. 1), “engenharia de requisitos é o processo pelo qual os requisitos de um produto de *software* são coletados, analisados, documentados e gerenciados ao longo de todo o ciclo de vida do *software*”.

Figura 6 – O processo de Engenharia de Requisitos.



Fonte: Adaptado de Sommerville (2003).

3.1 ELICITAÇÃO DE REQUISITOS DE *SOFTWARE*

Conforme Sommerville (2003) é na fase de elicitação e de análise dos requisitos que os engenheiros de *software* trabalham com a participação do solicitante e com os usuários finais do sistema. É nesse momento, que são identificados o domínio da aplicação, os serviços que serão fornecidos, e o desempenho aguardado para o sistema e as restrições de *hardware.*

Ele afirma também, que nessa fase são envolvidas muitas pessoas e grupo de pessoas da organização que solicita o *software*, denominadas como *stakeholders*, os quais podem deixar comprometida a compreensão e a elicitação dos requisitos quando:

a) não definirem quais processos o sistema de computador deverá conter, analisando apenas os termos gerais; e tiverem dificuldades de articular o que desejam, fazendo, por vezes, solicitações surreais, ignorando os custos de seus requisitos;
b) utilizarem-se de termos próprios para expressar os requisitos, dificultando o trabalho do engenheiro caso ele não conheça o domínio do cliente. Por isso, para que os requisitos solicitados sejam atendidos, é de suma importância que o engenheiro responsável conheça a área de negócio do cliente;
c) expressarem diferentemente requisito. Quando isso ocorre, o

engenheiro deve conhecer todas as fontes potenciais de requisitos e descobrir os pontos em comum e conflituoso;

d) os requisitos do sistema sofrerem influências dos fatores políticos;
e) houver dinamismo na forma de análise do ambiente econômico e de negócio, sendo este alterado inevitavelmente de acordo com a evolução do processo de análise, podendo surgir, dessa forma, novos requisitos solicitados por *stakeholders* diferentes.

Cada organização terá seu modelo geral particular que dependerá, sobretudo, do fator de negócio e de dados como nível de conhecimento da equipe envolvida, tipo de sistema que será implementado e os padrões que o mesmo terá. As atividades de processo são tratadas de forma espiral, isto é, são intercaladas à medida que o processo progride da parte interna para a externa da espiral.

Sommerville (2003) menciona ainda, que é considerada atividade de processo, o item abaixo apresentado:

a) obtenção de requisitos: neste processo, são reunidas todas as informações do sistema proposto. Com elas, obtêm-se os requisitos de usuário e de sistema. Tais informações são coletadas por meio de entrevistas com os *stakeholders*, nas quais conta-se com o auxilio de um cenário e do protótipo na obtenção dos requisitos. A abordagem oriunda da orientação feita na utilização do ponto de vista organiza o processo de elicitação e os próprios requisitos. Um ponto forte dessa abordagem é a possibilidade de se gerar um *framework* que venha a descobrir todos os conflitos entre os requisitos propostos por diferentes *stakeholders*. Os tipos genéricos de ponto de vista são:
   - ponto de vista de interação: é representado por pessoas ou por sistemas que irão interagir diretamente com o sistema a ser desenvolvido,
   - pontos de vista indiretos: são representados pelos *stakeholders* que não utilizam o sistema, mas que influenciam diretamente na criação dos requisitos,
   - pontos de vista de domínio: são as características e restrições de domínio que têm influência nos requisitos de sistema.

Esses pontos de vista apresentam requisitos distintos: o ponto de vista de interação disponibiliza os requisitos detalhados de característica e de interface do

sistema, já o ponto de vista indireto fornece os requisitos e as restrições da organização de alto nível e o ponto de vista de domínio concede os requisitos de restrições de domínio que se aplica no sistema.

Sommerville (2003) afirma que, além dos pontos de vista mencionados acima, devem também ser identificados os pontos de vista mais específicos, sendo os seguintes:

a) fornecedor e receptor de serviços do sistema: sistemas que devem interfacear diretamente com o sistema especificado; regulamentos e padrões que se aplicam ao sistema; fontes de requisitos de negócio e não funcionais do sistema; pontos de vista de engenharia que refletem os requisitos de pessoas que devem desenvolver, gerenciar, e manter o sistema; pontos de vista de marketing; outros pontos de vista que geram requisitos de características do produto, esperadas pelos clientes; e como o sistema deve refletir a imagem externa da organização;
b) classificação e organização de requisitos: atividade que envolve os requisitos não estruturados, agrupando os requisitos relacionados e os organizando em conjuntos coerentes;
c) priorização e negociação de requisitos: atividade responsável por resolver os conflitos de requisitos existentes, pois, quando vários *stakeholders* participam do processo, conflitos de requisitos são gerados, então são relacionados os requisitos por ordem de prioridade e entra-se em negociação;
d) documentação de requisitos formal ou informalmente.

3.2 ESTUDO DE VIABILIDADE

De acordo com Sommerville (2003) o processo de engenharia de requisitos deve ser iniciado com o estudo da viabilidade, onde consiste em um conjunto prévio dos requisitos de negócio ou um esboço das definições do sistema e de qual forma que o mesmo apoiar os processos de negócios do solicitante. O estudo da viabilidade resultará em um resumo em que terá a informação de que válido dar continuidade aos projetos de requisitos e ao processo de desenvolvimento. Este estudo mencionado acima é realizado de forma breve e atenciosa para se obter importantes respostas de

questões ilegíveis.

3.3 VALIDAÇÃO DE REQUISITOS

Conforme Pressman (2006) escreveu, a validação de requisitos faz a revisão da especificação para garantir que todos os requisitos do *software* sejam declarados sem que exista dúvida, para que seja realizada a correção das inconsistências, omissões e erros que o produto apresente, adequando-o às normas estabelecidas para o processo, para a tarefa, para o projeto e, sobretudo, para ele próprio.

Segundo Sommerville (2003), a validação coloca-se sobre à análise, pois a primeira é específica para a descoberta de erros ou inconsistências e problemas afins, a fim de se evitar o retrabalho no momento do desenvolvimento ou quando for operar no sistema, fator que geraria custos em excesso. O custo de correção de um requisito é mais alto que a correção de erros de um projeto e ou de uma codificação qualquer. O autor também afirma que, durante o processo de validação, as seguintes verificações devem ser realizadas, conforme menciona-se abaixo:

a) validade, na qual são verificadas as funções adicionais e diferentes que são necessárias;
b) consistência, ponto em que são verificadas as restrições ou descrições contrárias para a mesma função do sistema;
c) completeza, a qual consiste na verificação da documentação a fim de certificar-se deque todos os requisitos das funções e das restrições desejadas pelo usuário estejam especificados;
d) realismo, realiza com o intuito de se verificar a real possibilidade dos requisitos serem implementados de acordo com o conhecimento da tecnologia existente, levando-se em conta o orçamento e os prazos de desenvolvimento do sistema;
e) facilidade de verificação, na qual é analisado se os requisitos estão bem descritos a fim de que sejam diminuídas as divergências entre cliente e fornecedor, ou seja, devem-se descrever os conjuntos de testes que possam demonstrar que o sistema entregue atende a cada requisito.

Para as verificações acima serem aplicadas, há duas técnicas de revisões de requisitos, que são responsáveis pela análise sistemática de cada requisito. A

prototipação, já mencionada anteriormente é a criação de um modelo executável do sistema, que é apresentado e liberado aos usuários finais e aos solicitantes, para verificar se ele atente às necessidades reais do solicitante. E também a geração de casos de teste é a fase de teste de todos os requisitos.

3.4 DESENVOLVIMENTO DE REQUISITOS

O desenvolvimento de requisito é uma das etapas mais importantes dentro de um projeto de *software*, pois se um requisito for mal desenvolvido, muitos problemas futuros podem ocorrer no decorrer do processo de desenvolvimento. Faz-se necessária a utilização de alguns passos básicos para que os requisitos sejam bem desenvolvidos, sendo os seguintes:

a) o workshop inicial é primeiro contato, por meio do qual se obtém a primeira visão que a organização tem;

b) as entrevistas com *stakeholders* (usuários) são realizadas com funcionários e/ou usuários de cada setor da organização, documentando-se todas informações obtidas;

c) a validação do processo de negócio por setor consiste na avaliação que os usuários realizam acerca do modelo de processo de negócio de seu setor. Caso haja correções, estas serão realizadas. Apos a aprovação dos usuários, as assinaturas são coletadas, servindo de comprovação de que os modelos estão de acordo com o solicitado;

d) a análise e a validação do processo de negócio ocorre após a aprovação dos modelos de negócio dos setores. Os modelos são unificados e, juntamente com a diretoria da organização, é realizada a última avaliação;

e) a elicitação dos requisitos consiste na identificação dos requisitos funcionais e não funcionais, registrando-os na ferramenta Enterprise Architect (EA);

f) a modelagem dos casos de uso é o agrupamento dos requisitos de acordo com a funcionalidade, originando-se, dessa forma, o modelo de caso de uso;

g) a criação do protótipo é uma ação paralela à modelagem dos casos de uso. Por meio dela, são criados os protótipos das telas dos sistemas,

para a posterior validação do usuário.

3.4.1 Uma Visão dos Modelos Clássicos

Sommerville (2003) afirma que, os requisitos devem ser escritos em linguagem natural, pois eles devem ser entendidos por pessoas que não possuem tais conhecimentos técnicos. Quanto mais bem detalhado os requisitos estiverem, a compreensão será mais fácil. Exemplificando melhor, existem alguns modelos de sistemas que são baseados em abordagens distintas durante o processo de análise dos requisitos. Para as abordagens diferentes são descritos alguns modelos:

a) fluxo de dados: neste modelo, é mostrada a maneira como os dados serão processados nas diferentes etapas do sistema;
b) composição: é o modelo que representa a forma como são compostas a composição ou a agregação das entidades do sistema;
c) arquitetura: este modelo apresenta os principais subsistemas do projeto;
d) classificação: neste modelo, os diagramas de objeto/herança mostram as características comuns das entidades;
e) estímulo-resposta: por meio deste modelo, é apresentado como o sistema reage aos eventos externos e internos.

3.4.1.1 Listagem dos requisitos

Portella (2011) leciona que, a listagem de requisitos é o documento em que estão presentes todos os requisitos dos sistemas, que foram levantados pela equipe responsável, conforme citado anteriormente. Onde esta equipe faz as entrevistas, o acompanhamento e as reuniões na organização solicitante, contando com as participações dos funcionários e usuários.

Nesta documentação, são mencionados os requisitos funcionais e não funcionais a que serão implementados no decorrer do projeto. Considerando desta forma que um requisito funcional define o que o sistema faz, é descrito as ações de transformação que os componentes de *software* ou hardware devem exercer sobre as entradas, no intuito de obter as saídas. Já os requisitos não funcionais especificam os atributos do sistema durante a sua execução e determinam também como os atributos definidos por eles podem ser avaliados como atributos de qualidade ou de restrições

de sistemas de *softwares* ou de processos de *software*.

3.4.1.2 Modelos de casos de uso

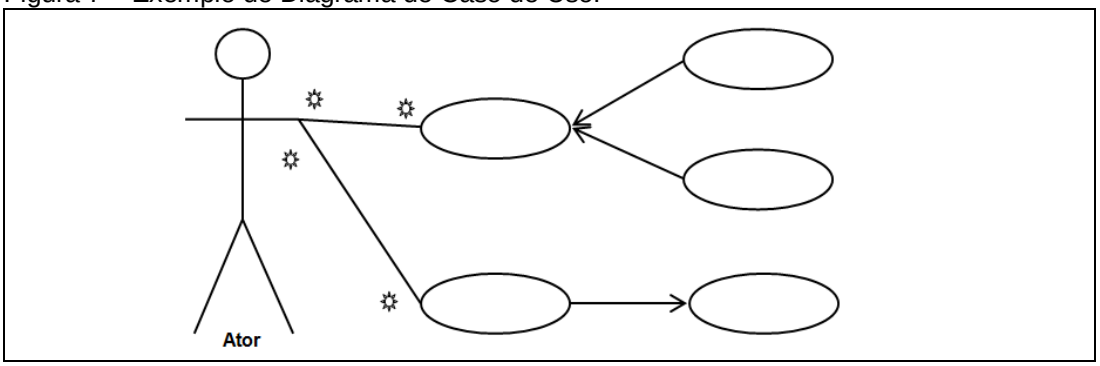
Modelos ou diagramas de caso de uso sinalizam a funcionalidade fornecida pelo sistema.

De acordo com Savi (2009), o diagrama de caso de uso mostra como o sistema vai interagir com os usuários (pessoas ou outros sistemas). Sendo assim, diagramas de caso de uso descrevem as interações e funcionalidades entre as categorias de usuários e o sistema.

Pressman (2006, p. 153) afirma que “descrevem uma sequência de ações que são realizadas por um ator à medida que o ator interage com o *software*”.

Entende-se que a finalidade principal deste diagrama é de auxiliar as equipes de desenvolvimento a visualizar as exigências funcionais de um sistema. Neste diagrama, os atores se relacionam com os processos essenciais, assim como os relacionamentos entre casos diferentes de uso. Este diagrama nos mostra grupos de casos de uso, onde os seres humanos interagem com o sistema, ou até outros sistemas interagem com o mesmo. Observa-se um exemplo de diagrama de casos de uso.

Figura 7 – Exemplo de Diagrama de Caso de Uso.



Fonte: Costa (2001 apud Junqueira, 2006).

Os casos de uso são utilizados, para manter a funcionalidade desejada do sistema e realizar testes para a comunicação dos participantes, e por fim, a validação do sistema. Em um projeto, cada participante tem a sua visão e para o sucesso, todos devem interagir falando a mesma linguagem, focando o mesmo objetivo. A Linguagem de Modelagem Unificada (UML) com o diagrama de caso de uso vem

tentar exatamente isso, dar uma visão funcional do sistema como um todo e todas as interações possíveis, trazendo benefícios conforme apresentado no quadro 10.

Quadro 10 – Benefícios de Casos de Uso.

Benefícios conforme Leffingwell	Os Casos de Uso são relativamente fáceis de escrever e mais fáceis de ler.
	Forçam os desenvolvedores a pensar no projeto de um sistema da perspectiva de um usuário.
	Dão contexto para os requisitos do sistema. Pode-se compreender porque uma exigência existe e como o sistema chega aos seus objetivos.
	É uma ótima ferramenta no processo da análise, ajudando a compreender o que o sistema necessita fazer e como realizar isso.
	É uma ferramenta que atua no projeto e no desenvolvimento. Reduz o risco de erros ao se implementar os requisitos do sistema, por dar uma visão geral de todos os requisitos e onde cada um deve chegar.
	É útil também no processo de teste, ajudando a definir se o sistema chegou realmente onde deveria chegar.

Fonte: Adaptado de Leffingwell (2003 apud Ferrari, 2006).

O descritivo dos casos de uso provê informações que ajudam ao se especificar as propriedades das classes necessárias para atuar como um caso de uso, representando as funcionalidades de alto nível do sistema, detalhando o que o sistema deveria executar, isto é, fazer.

Conforme o autor Ivar Jacobson (2002), o caso de uso pode ser definido como um documento narrativo que descreve a seqüência de eventos de um ator que usa um sistema para completar um processo.

3.4.2 Uma Visão dos Modelos Ágeis

Sommerville (2003) diz que os métodos ágeis (já citados no capítulo 2) se iniciaram nos anos 80 e início da década de 90. A visão deste modelo era de qualificar a maneira de obter o melhoramento dos *softwares*. Para isso, o mesmo defendia a utilização dos conhecidos planejamentos de projeto, garantindo a qualidade formalizada com o uso de métodos de análise e projetos apoiados por ferramentas de *CASE* e por rigorosos controles de processo de desenvolvimento.

O autor também afirma que os métodos ágeis têm alguns princípios básicos: o envolvimento do cliente no processo de desenvolvimento, pois o sucesso depende da participação dos usuários no processo de desenvolvimento. Outro é a participação de membros individuais que não estão compondo a equipe de desenvolvimento. Quando existe a participação de usuários no processo de desenvolvimento pode ocasionar um desconforto para a equipe pois inicia-se a

pressão para a entrega do sistemas no prazo determinado e assim se compromete o sistema desenvolvido.

3.5 MODELAGEM DE SISTEMAS DA INFORMAÇÃO

A modelagem de sistemas da informação se inicia por uma associação da maneira de perceber os fenômenos ao modo de pensamento que predomina em cada indivíduo, na observação de certo fenômeno, em momento específico do tempo. O Sistema é definido então, como uma coletânea de estruturas e recursos que são interagidos segundo uma lógica de tal forma para alcançar um ou mais objetivos, sendo eles específicos ou não. Para Silva (2005, p. 2), em seu artigo que publicou na internet, “os estudos destes sistemas podem dar-se sob diferentes formas de abordagem”.

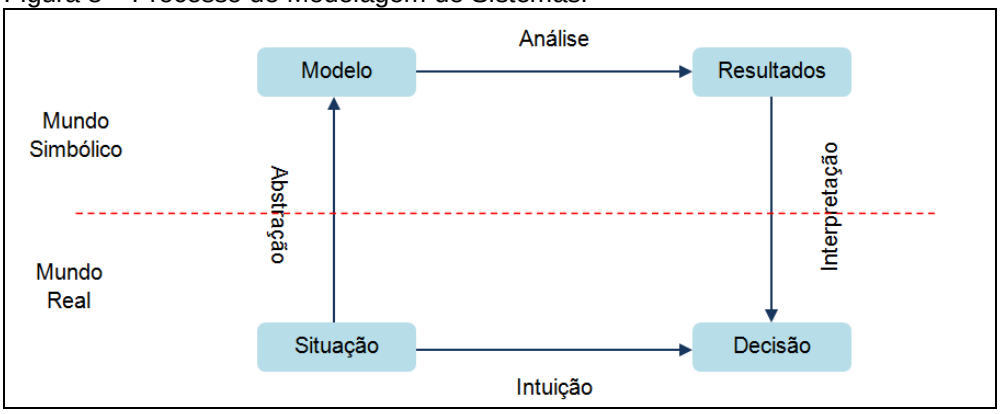
Quadro 11 – Formas de Abordagem.

Formas de abordagem		
Formas	**Abordagem**	**Resultado**
1ª	Refere-se na intervenção direta sob-rotinas operacionais promovendo implementações e, ou, alterações de procedimentos até que sejam obtidas as condições ideais.	Estas ações fazem requerer do tomador de decisão a condução de estudos preliminares e experiência, para que as alterações não minorem a performance do sistema.
2ª	Refere-se à utilização de modelos que representem os sistemas reais. Os modelos podem apresentar-se como protótipos ou como modelos matemáticos.	Este presta-se à soluções analíticas, como por exemplo um modelo de regressão, ou a simulação, permitindo assim, reconstituir a rotina funcional de um dado sistema real.

Fonte: Adaptado de Silva (2005).

Pode ser considerado um sistema de *software* bem sucedido, um sistema que é capaz de atender às necessidades dos usuários com qualidade. Para implementar tal sistema de *software*, com eficiência e eficácia de recursos e de maneira previsível, a modelagem de sistemas é utilizada. A figura 8 deixa mais visível este processo de modelagem de sistemas.

Figura 8 – Processo de Modelagem de Sistemas.



Fonte: Adaptado de Gomide (2007).

Conforme consta na figura 8, a abstração e a interpretação são decisivas para o processo de modelagem para a tomada de decisão. Neste âmbito a modelagem de *software* serve para auxiliar na organização das informações, criando um escopo para o projeto que mostra a real utilidade perante o sistema, considerando as análises de requisitos, desta forma o *software* terá sua validação em seu conjunto.

4 WEB SERVICES

Segundo Potts (2003, p.3), “um Web Service é uma aplicação de *software* que pode ser acessada remotamente usando diferentes linguagens baseadas em XML”.

A promessa dos Web Services é baseada na interoperabilidade, ou seja, toda aplicação de *software* no mundo pode se comunicar com qualquer outra. Ultrapassando todas as antigas barreiras de local, sistema operacional, linguagem, protocolo, entre outros.

Os Web services se baseiam no envio de mensagens e*Xtensible Markup Language (*XML) em um formato *Simple Object Access Protocol* (SOAP) específico. Mas como a interoperabilidade dos Web Service seja difícil de ser alcançada, as organizações especificadoras buscam a definição de padrões para atingir esta meta.

4.1 PADRONIZAÇÃO

Estes padrões são úteis, pois a indústria pode não aderir a projetos criados por uma única empresa, sendo que podem ficar atrelados a um único fornecedor, e se este não abrir o padrão desenvolvido, a tecnologia pode não evoluir e os investimentos escoarem pelo ralo. Por isso, os concorrentes se unem na organização das especificações para criarem padrões de consenso e com maior probabilidade de obtenção de credibilidade perante a indústria.

Dentre as principais organizações especificadoras, a *World Wide Web Consortium* (W3C) controla as especificações SOAP, WSDL, XML, XML Schema e *HyperText Transfer Protocol* (HTTP) e a *Organization for the Advancement of Structures Information Standards* (OASIS) controla as especificações *Universal Description, Discovery and Integration* (UDDI), WS-Security e *Security Assertion Markup Language* (SAML).

Potts (2003, p. 284) registra, “Se um padrão demorar a ficar pronto, os fornecedores de *softwares* implementarão suas próprias soluções, tornando, assim, mais difícil para que eles adotem o novo padrão posteriormente”. Depois que estes padrões são editados ou revisados e então publicados, os fornecedores de ferramentas as implementam em seus produtos. A seguir uma relação de ferramentas para criação de Web Services, mais conhecidas: Apache Axis, Java, Visual

Studio.NET, Web Services.*NET Clients*, BEA *WebLogic WorkShop*, IBM *WebSphere Studio Application Developer*.

Potts (2003, p. 10), “Com o inimigo a bordo, o crescente grupo de apoiadores procura uma organização especificadora, como W3C ou a OASIS, para administrá-lo. [...], o comitê publica a especificação e a chama de padrão (OASIS) ou de recomendação (W3C)”.

4.2 ARQUITETURA DOS WEB SERVICES

Os Web Services são aplicações de *softwares* que pode ser acessadas remotamente, usando XML na sua estrutura de programação.

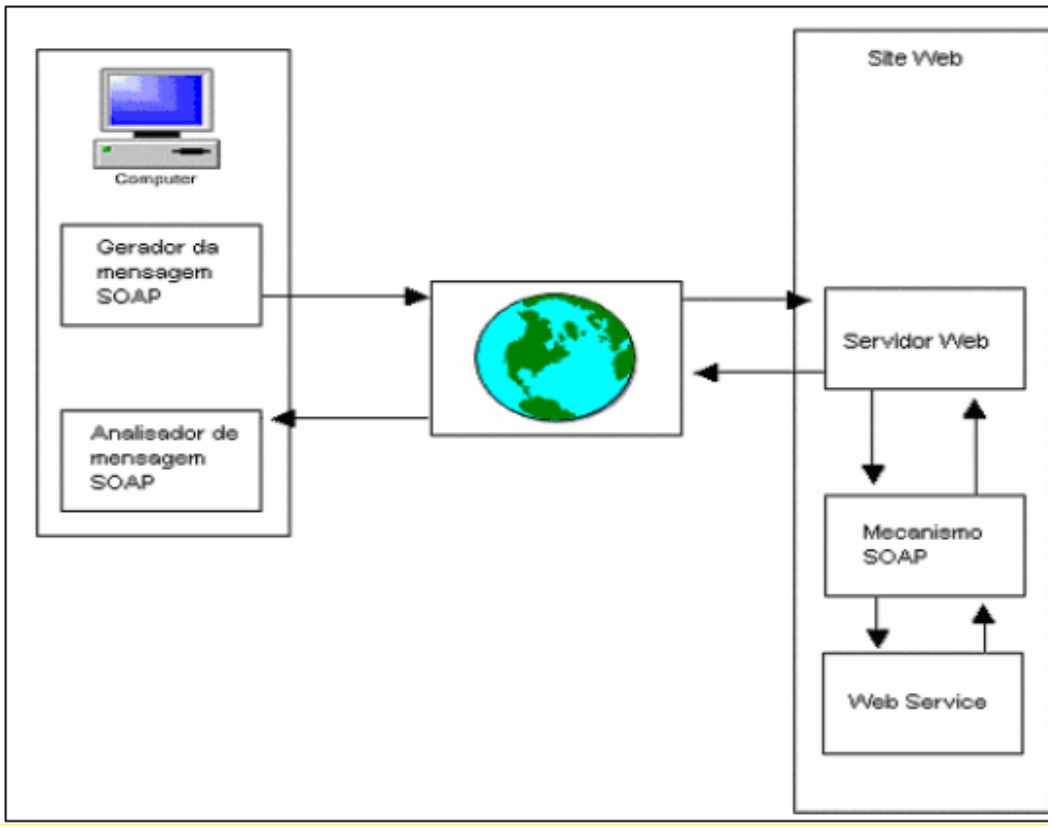
Geralmente são identificados por uma Uniform Resouce Locator (URL), como qualquer página da Internet. A diferença está no conteúdo do que é enviado na requisição do cliente para servidor (POTTS, 2003).

Um Web Service é um conjunto de funções que podem ser invocados através da rede, utilizando o protocolo SOAP. Estas funções recebem o nome de web methods. Eles permitem que dois programas se comuniquem de uma maneira tecnicamente muito semelhante à invocação de páginas Web.

Web Services promete realizar com o SOAP, que clientes e servidores heterogêneos possam compartilhar aplicações usando módulos que são descritos, publicados, localizados e invocados através de uma rede de forma transparente. Neste ponto surge uma interface para que uma aplicação possa usar a Internet, e não um usuário. Onde aplicativo qualquer possa fazer a mesma coisa, isso consiste a ideia de Web Services (CZERVENY, 2004).

A arquitetura e o fluxo que as informações irão percorrer poderão ser observados na figura 9.

Figura 9 – Fluxo dos dados na arquitetura Web Service.



Fonte: Potts (2003, p.5).

Um Web Service pode ser considerado como um serviço disponível na WEB, porém sem telas gráficas, que utilizam chamadas e retornos em formato XML, no caso utilizando padrão SOAP.

O cliente que utiliza este padrão de distribuição precisa de um arquivo conhecido como Web Service Description Language (WSDL), onde contém as regras para a comunicação. Logo após a obtenção das informações do arquivo, o programador do lado cliente poderá montar as classes referentes ao serviço que pretende acessar, e assim criar o seu sistema que acessará e receberá tais informações (PAULON, 2004).

O Desenvolvimento das aplicações é divido então em dois: o servidor e o cliente.

No servidor, onde serão implementados e disponibilizados os serviços, ou seja, todas as regras de funcionalidades dos objetos serão desenvolvidas, e é aqui que o documento WSDL será armazenado, pois contém as bases para se estabelecer à conexão.

No lado cliente, onde o usuário estará em contato com os recursos

disponibilizados pelos objetos que estão no servidor, pode ser escrito em qualquer linguagem de programação. Mas deverá seguir as especificações do documento WSDL.

Existem dois tipos de troca de informação entre o cliente e o servidor e se diferencia pelo lado que iniciou a comunicação:

a) requisição/resposta: o cliente inicia a ação fazendo uma requisição ao servidor;
b) solicitação/resposta: o servidor realizar a primeira ação solicitando uma resposta do cliente.

Simplificando, Web Service é a maneira dos aplicativos se comunicarem e trabalhar juntos via um dispositivo de comunicação ou rede como Intranet e principalmente a Internet. As entidades que interagem são as seguintes:

a) solicitadores: geralmente o lado cliente. Uma aplicação que deseja receber serviços de uma outra aplicação. O Cliente não sabe de onde está vindo às mensagens, nem como localizá-la, para isso eles recorrem aos agenciadores que requisita uma operação de busca;
b) provedores: lado servidor que irá disponibilizar os serviços, possibilitando aos clientes, independentemente de onde se encontram, acessá-los e usufruir alguns dos seus benefícios;
c) agenciadores: peça de fundamental importância irá estabelecer o sincronismo de acesso dos dados entre os lados. O servidor retorna as informações sobre os serviços em resposta a um critério de pesquisa que tenha sido submetido por um solicitador. Essas informações com base em classificações que correspondiam ao critério de pesquisa retornam para o Web Service.

Em um momento de comunicação, um provedor de serviços irá publicar a disponibilidade a que se propõem e responderá a requisições do que a mesma tenha publicado. A operação de publicação permitirá ao provedor registrar suas habilidades e seus requisitos de interface junto a um agenciador de serviços. E este por sua vez, registrará e irá categorizar os provedores e irá oferecer esses serviços de pesquisa. A operação de busca permitirá que o solicitador encontre a tarefa desejada junto ao agenciador.

5 EXIGÊNCIAS DA ANTT – RESOLUÇÃO Nº 3.658/11

A Agência Nacional de Transportes Terrestres, necessitando garantir a movimentação de bens em cumprimento a padrões de eficiência e a modicidade nos fretes no Brasil, e também levando em consideração os problemas causados no mercado de transporte rodoviário de cargas devido às sistemáticas ineficientes de pagamento de frete, decidiu através do artigo 1º da resolução nº 3.658/11 (BRASIL, 2011) regulamentar o pagamento do valor do frete referente à prestação dos serviços de transporte rodoviário de cargas, onde já está previsto no artigo 5º-A da lei nº 11.422, de 2007.

Segundo o artigo 2º desta resolução considera-se:

> I - Operação de Transporte: viagem decorrente da prestação do serviço de transporte rodoviário de cargas por conta de terceiros e mediante remuneração.
> II - Código Identificador da Operação de Transporte: o código numérico obtido por meio do cadastramento da Operação de Transporte nos sistemas específicos;
> III - Contrato de Transporte: as disposições firmadas, por escrito, entre o contratante e o contratado para estabelecer as condições para a prestação do serviço de transporte rodoviário de cargas por conta de terceiros e mediante remuneração;
> IV - contratante: a pessoa jurídica responsável pelo pagamento do frete ao Transportador Autônomo de Cargas – TAC ou a seus equiparados, para prestação do serviço de transporte rodoviário de cargas, indicado no cadastramento da Operação de Transporte;
> V - contratado: o TAC ou seu equiparado, que efetuar o transporte rodoviário de cargas por conta de terceiros e mediante remuneração, indicado no cadastramento da Operação de Transporte;
> VI - subcontratante: o transportador que contratar outro transportador para realização do transporte de cargas para o qual fora anteriormente contratado, indicado no cadastramento da Operação de Transporte;
> VII - consignatário: aquele que receberá as mercadorias transportadas em consignação, indicado no cadastramento da Operação de Transporte ou nos respectivos documentos fiscais;
> VIII - proprietário da carga: o remetente ou o destinatário da carga transportada, conforme informações dos respectivos documentos fiscais;
> IX - administradora de meios de pagamento eletrônico de frete: a pessoa jurídica habilitada pela ANTT, responsável, por sua conta e risco, por meio de pagamento eletrônico de frete aprovado pela ANTT.

De acordo com o artigo nº 3, equiparam-se ao Transportador Autônomo de Cargas (TAC), a empresa de transporte rodoviário de cargas que possuir, em sua frota, até três veículos registrados no Registro Nacional de Transportadores de Cargas, e as Cooperativas de Transportes de Cargas, sendo que para fins de comprovação da quantidade de veículos, será considerada a frota da Empresa de

Transporte Rodoviário de Carga (ETC) na data de cadastramento da Operação de Transporte ou, na sua ausência, na data de início da viagem.

No 4º artigo desta resolução criada por Brasil (2011), são mencionados os meios que podem ser pagos ao TAC ou a seu equiparado quando os mesmos prestarem serviço de transporte rodoviário de cargas.

A partir do 5º artigo são impostas todas as regras específicas para o cadastramento da operação de transporte.

Conforme publicado no portal Dbtrans (2011) a antiga modalidade de pagamento de denominada carta-frete era uma ordem de pagamento emitida pelos contratantes de caminhoneiros autônomos como forma de pagamento do frete realizado. De posse do documento, o caminhoneiro procurava um posto de combustíveis que aceitasse trocar a carta-frete por dinheiro em troca de consumo de diesel. Neste cenário os postos de combustíveis atuavam como bancos movimentando cerca de 60 bilhões de reais anuais à margem do sistema formal.

Conforme artigo 37 da resolução criada, a resolução nº 3.658/11 entrou em vigor a partir de sua data de publicação, que ocorreu em 22 de janeiro de 2012. Desta forma, a partir desta data ficou vedada a utilização de carta-frete, bem como de qualquer outro meio de pagamento não previsto na resolução para fins de remuneração do transportador autônomo ou de seus equiparados, decorrente da prestação do serviço de transporte rodoviário de cargas por conta de terceiros e mediante remuneração.

A comprovação do cumprimento da resolução diante da fiscalização ocorre através da apresentação do Código Identificador da Operação de Transporte (CIOT), onde este código é obtido através de integração via Web Service com a Administradora de Pagamento Eletrônico de Frete.

5.1 ADMINISTRADORAS DE PEF

Após regulamentar a resolução que padronizava o pagamento do frete referente à prestação de serviços de transporte rodoviário de cargas, a Agência Nacional de Transportes Terrestres passou a homologar administradoras de meios de pagamento de eletrônico de frete (BRASIL, 2011).

As administradoras de pagamento eletrônico de frete são as empresas habilitadas pela ANTT, responsáveis por realizar o trâmite de forma segura e

eletrônica, do pagamento referente a serviço de transporte rodoviário que o contratante faz ao contratado responsável por realizar o frete.

Conforme indica a página web oficial da ANTT, existem atualmente cerca de 20 Administradoras de Meios de Pagamento Eletrônico de Frete habilitadas pelo órgão.

5.1.1 Particularidades e modelos de layouts das administradoras PEF

Mesmo sendo permitido utilizar linguagens de programação diferentes nas "pontas de uma integração" via Web Service, o proprietário do Web Service consumidor dos XMLs estabelece modelos específicos de acordo com as regras de seu negócio. Foi desta forma, que as administradoras de PEF disponibilizaram web services com modelos de integração (layouts) que possuem particularidades diferentes uma das outras.

Devido à existência de particularidades em cada administradora de pagamento eletrônico de frete atualmente há uma grande preocupação com o tempo gasto no desenvolvimento e atualizações de *softwares* que integram com estas administradoras de PEF. Desta forma evitando-se a utilização de tempo em excesso e recursos muitas aplicações não possuem um padrão de integração com as administradoras e na maior das vezes não reutilizam código.

Realizando um comparativo de alguns aspectos da documentação das administradoras de pagamento eletrônico de frete 1, 2 e 3, respectivamente Dbtrans, Pamcary e Apisul obtém-se algumas definições.

a) no comparativo de requisitos técnicos de integração do PEF mostrada no quadro 12 não foram encontrados requisitos técnicos no manual da administradora Dbtrans:

Quadro 12 – Comparativo de requisitos técnicos de integração PEF.

Funcionalidade	Requisitos técnicos
Administradora PEF 1	Não divulgado.
Administradora PEF 2	Somente definiu requisitos para utilização do adaptador, sendo os seguintes: - Windows – acima de 2000; Acesso liberado para internet; - Java Virtual Machine instalado (qualquer versão acima de jre-1_6).
Administradora PEF 3	- Sistema operacional Windows versão superior a 2000; - Acesso internet (navegador Internet Explorer 7 ou superior,

	Chrome e Firefox);

Fonte: Do autor.

b) na comparação dos aspectos de segurança todas as administradoras mostradas no quadro 12 possuem particularidades:

Quadro 13 – Comparativo de segurança na integração PEF.

Funcionalidade	Segurança
Administradora PEF 1	Todo o tráfego de dados entre a estação de trabalho dos clientes, rede credenciada e o portal da administradora é criptografado, antes do envio das informações via Internet, através de conexão segura (Protocolo HTTPS), certificada pela autoridade certificadora VeriSign (www.verisign.com). Esta medida garante que os dados trafegados não serão capturados, ou mesmo alterados no seu curso.
Administradora PEF 2	A comunicação entre cliente e servidor será feita através do SOAP. Esse protocolo é definido em XML. Para transporte das mensagens será usado o HTTPS. O acesso ao servidor que hospeda Web Service Pamcary se dará por acesso à internet. O cliente deverá possuir um certificado ICP-BRASIL e um usuário correspondente no Sistema Pamcard com perfil específico para realizar as transações. Este usuário somente poderá ser utilizado nestas transações, não sendo permitido a este usuário acessar o sistema na Web.
Administradora PEF 3	O acesso é restrito aos endereços de IPs fornecidos pelos clientes. Caso o cliente altere o IP de seus servidores, o mesmo deverá informar com antecedência para que o acesso não seja bloqueado.

Fonte: Do autor.

c) no quesito disponibilidade somente a Dbtrans publicou em suas documentações:

Quadro 14 – Comparativo de disponibilidade na integração PEF.

Funcionalidade	Disponibilidade
Administradora PEF 1	A proposta de alta disponibilidade de acesso ao portal da administradora consiste em manter ambientes redundantes com balanceamento de carga entre os vários servidores que hospedam o sistema, assim como acompanhar e analisar as ferramentas de monitoramento de controle de todo o ambiente, físico e lógico, a fim de garantir os níveis de serviço exigidos.
Administradora PEF 2	Não divulgado.
Administradora PEF 3	Não divulgado.

Fonte: Do autor.

d) referente aos métodos disponibilizados via Web Service a Apisul é a administradora que possui menos serviços:

Quadro 15 – Comparativo de métodos disponibilizados via WS na integração PEF.

Funcionalidade	Métodos disponibilizados via WS

Administradora PEF 1	Autenticar cliente; Manter Viagem; Manter Motorista; Manter Transportador; Manter Veículo; Consultar Rotas; Detalhar Rota; Consultar Tarifas; Consultar Viagens; Cancelar Viagem; Retificar Viagem; Detalhar Viagem; Consultar Cláusulas; Consultar Estabelecimentos; Registrar Operação Viagem; Manter Operação Viagem;
Administradora PEF 2	Consultar Cartão; Consultar Favorecido; Consultar RNTRC; Consultar Frota; Roteirizar; Consultar Conta; Incluir Cartão Portador Frete; Incluir Favorecido; Incluir Conta; Inserir Remetente / Destinatário; Incluir Contrato Frete; Consultar Status da Parcela; Consultar Contrato de Frete; Alterar Status da Parcela; Consultar Status do Pedágio; Alterar Status do Pedágio; Cancelar Viagem / Contrato de Frete; Alterar Contrato de Frete; Atualizar Valores do Contrato de Frete; Encerrar Contrato de Frete.
Administradora PEF 3	Declaração de Operação de Transporte; Encerramento de Operação de Transporte; Retificação de Operação de Transporte; Cancelamento de Operação de Transporte;

Fonte: Do autor.

e) das três administradoras somente a Pamcary realiza a autenticação com certificado digital:

Quadro 16 – Comparativo de certificado digital na integração PEF.

Funcionalidade	Certificado digital
Administradora PEF 1	Não
Administradora PEF 2	Sim
Administradora PEF 3	Não

Fonte: Do autor.

f) na verificação da existência de mais algum tipo de comunicação específica que as administradas utiliza além de Web Service, conclui-se que somente a Pamcary possui um outro meio de integração:

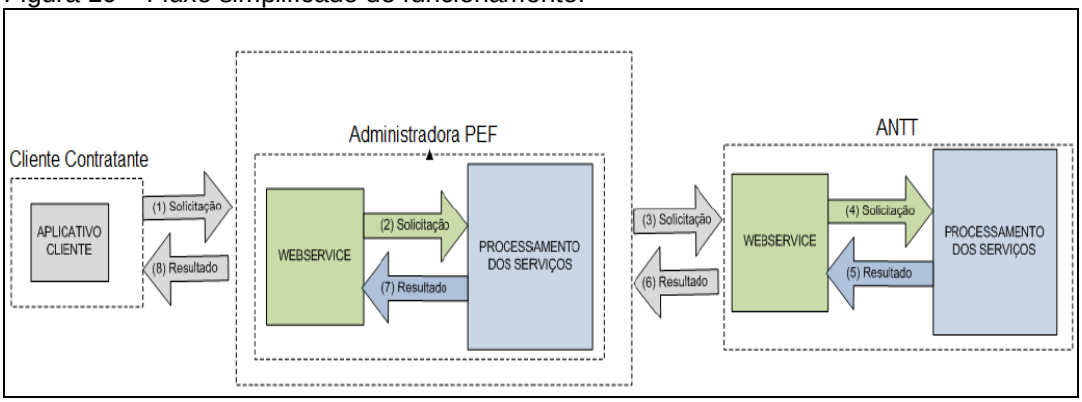
Quadro 17 – Comparativo de comunicação específica além de WS.

Funcionalidade	Comunicação específica além de WS.
Administradora PEF 1	Não.
Administradora PEF 2	Sim, além de disponibilizar comunicação via Web Service para troca de informações em tempo real, possui um adaptador auxiliar (executável) para envio e retorno de XMLs.
Administradora PEF 3	Não.

Fonte: Do autor.

Apesar das administradoras PEF possuírem modelos de documentação e web services diferentes uma da outra, o fluxo de integração com a ANTT sempre segue um único padrão, conforme pode ser analisado na figura 10.

Figura 10 – Fluxo simplificado de funcionamento.



Fonte: Dbtrans (2011, p. 13).

Conforme apresentado na figura 10, o cliente final consome o Web Service da administradora de pagamento eletrônico de frete, e na seqüência a administradora consome alguns serviços do Web Service da ANTT, desta forma a Agência Nacional de Tranportes Terrestres devolve a requisição para a administradora e a mesma devolve a requisição ao cliente solicitante. Todas as comunicações ocorrem de forma quase instantânea, podendo ser imperceptíveis na visão do usuário final.

6 TRABALHOS CORRELATOS

Atualmente, os profissionais de engenharia de *software* realizam esforços para contribuições referentes a melhorias e soluções de problemas ligados a vários processos da área, desde o levantamento de requisitos até o desenvolvimento e implantação de *software*. Percebe-se os inúmeros artigos e monografias existentes que auxiliam e contribuem para esta boa prática.

Dispõe-se a seguir os trabalhos correlatos, com embasamento em engenharia de requisitos, aplicação de métricas de *software* e em melhorias através de sistemas que a atendem legislação de transporte e logística de distribuição de cargas.

6.1 ENSINO DA ENGENHARIA DE REQUISITOS POR MEIO DE UM AMBIENTE COLABORATIVO

A monografia de Bacharelado apresentada como requisito parcial para formação no curso de Ciência da Computação da Universidade de Brasília pelas acadêmicas Isabella Cristina Bueno e Sirley Guimarães Garrido visa à análise do processo de engenharia de *software* desde o processo de definição de requisitos até a entrega do *software* ao cliente. Dentre os pontos importantes da engenharia de *software* que as acadêmicas abordaram, foi centralizada no trabalho a engenharia de requisitos, propondo o estudo dela de mecanismos apropriados ao entendimento, à análise e à avaliação da viabilidade do produto que o cliente espera. Elas relatam que é também neste processo que surgem o maior número de variáveis problemáticas tais como manter o sucesso do desenvolvimento de *software*.

No trabalho também consta que na engenharia de requisitos, os problemas encontrados podem ser minimizados com uma adequada qualificação profissional dos responsáveis por essa área e que esta qualificação pode ser obtida por meio de aprimoramentos de técnicas, oferecido por instituições de ensino.

6.2 APLICAÇÃO DE MÉTRICAS DE *SOFTWARE* NO MÉTODO SCRUM DA METODOLOGIA ÁGIL

A abordagem deste tema provém do Trabalho de Conclusão de Curso (TCC) de Ciência da Computação do acadêmico Mateus Luiz Gamba na Universidade do Extremo Sul Catarinense (UNESC) defendido em 2009 que aborda um estudo de aplicação de métricas de *software*, realizado a fim de que fossem obtidos os prazos estimativos para o desenvolvimento de uma interação do método Scrum, utilizando-se a metodologia ágil. O autor descreve que no desenvolvimento de um projeto, a metodologia ágil propõe-se a contribuir na implementação rápida e flexível de qualquer mudança. Ele também descreve que é considerada uma das metodologias que melhor atende às necessidades básicas para a obtenção de um resultado significativo no processo de desenvolvimento de *software*. A metodologia elicitada pelo acadêmico também favorece no gerenciamento de projeto, pois oferece flexibilidade e é adaptável a constantes mudanças.

6.3 USO OTIMIZADO DE INFORMAÇÕES DO ATENDIMENTO NO POSTO DE QUELUZ

Em 2012, o acadêmico Cleber Rezende Carvalho da Faculdade de Tecnologia Prof. Waldomiro May, situada na cidade de Cruzeiro-SP abordou este tema para seu projeto de TCC. Este tema abordado pelo acadêmico enfoca no atendimento da legislação de pesagem de veículo no Brasil através de otimização de informações.

O objetivo geral do trabalho criado foi de atender a legislação vigente no que diz respeito à pesagem de veículos, através do desenvolvimento de um protótipo de esquema de comunicação entre três base de dados diferentes para suprimir redundâncias e redigitações e assim eliminar inconsistências e reduzir o tempo de atendimento no Posto de Pesagem de Veículos na cidade de Queluz/SP.

6.4 PROTÓTIPO DE *SOFTWARE* PARA LOGÍSTICA DE DISTRIBUIÇÃO

No ano de 2004, para obter o grau de bacharelado em Ciência da Computação da Universidade Regional de Blumenau, Santa Catarina (SC), a

acadêmica Viviane Bittencourt Rosa propôs este tema para seu Trabalho de Conclusão de Curso. A motivação da acadêmica neste trabalho foi de buscar possíves soluções para problemas logísticos que ocorrem constantemente na rotina de diversas empresas e transportadoras. A mesma relata que a resolução deste tipo de problema não é trivial, e que uma forma de contornar esta complexidade é utilizar uma técnica que tem apresentado bons resultados em problemas semelhantes. A técnica optada foi a de utilização da técnica de *Constraint Satisfaction Problem* (CSP), também chamada de programação de restrições, que viabiliza a especificação e a resolução de problemas semelhantes aos que foram mencionados aqui.

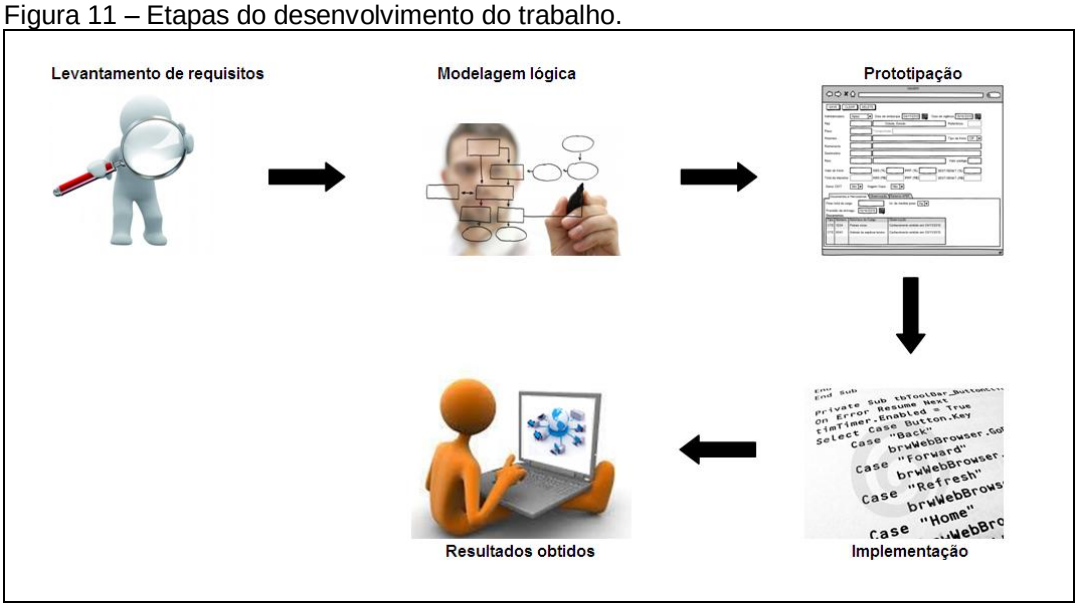
7 ABORDAGEM DA FERRAMENTA DE INTEGRAÇÃO

O trabalho corrente tem como objetivo realizar a abordagem de uma ferramenta para comunicação via Web Service com Administradoras de Pagamento Eletrônico de Frete, visando atender a resolução nº 3.658/11 disposta pela Agência Nacional de Transportes Terrestres.

7.1 METODOLOGIA

Conforme pode ser visualizado na figura 11, a metodologia do trabalho desenvolvido está disposta em 5 etapas, sendo as seguintes: levantamento de requisitos, modelagem lógica, prototipação, implementação e resultados obtidos.

Figura 11 – Etapas do desenvolvimento do trabalho.



Fonte: Do autor.

As etapas da engenharia de software apresentadas na figura 11 são essenciais para atingir os objetivos do trabalho proposto, e serão detalhadas no decorrer do trabalho desenvolvido, após a descrição dos recursos utilizados.

Para realização de cada etapa se fez necessário o estudo das ferramentas a serem utilizadas, onde as mesmas serão relacionadas na próxima seção (7.1.1).

7.1.2 Levantamento de Requisitos

Nesta primeira etapa do projeto executaram-se alguns passos para elicitação de todos os requisitos do processo de comunicação via Web Service com as Administradoras de Pagamento Eletrônico de Frete.

No primeiro passo foram solicitados os manuais atualizados das três administradoras mencionadas respectivamente no capítulo cinco (Dbtrans, Pamcary e Apisul), onde se definiu que o levantamento seria realizado com base nos manuais disponibilizados pelas administradoras Dbtrans e Apisul, devido às mesmas possuírem uma documentação mais acessível dos Web Services oferecidos, possibilitando assim uma porção ainda maior dos resultados obtidos deste trabalho.

Após definição das administradoras, realizou-se um estudo de quais serviços são necessários consumir em cada Web Service para atender a resolução disposta pela ANTT, visto que as administradoras disponibilizam serviços adicionais que não são necessários para atender a resolução 3.658/11. Ficou concluído que no mínimo devem ser utilizados os seguintes serviços para atendimento da resolução, conforme apresentados no quadro 17.

Quadro 17 – Métodos básicos para atendimento da resolução.

Administradora	**Métodos**
Dbtrans	Autenticar cliente; Manter Viagem; Manter Transportador; Manter Veículo; Consultar Rotas.
Apisul	Declaração de Operação de Transporte.

Fonte: Do autor.

A partir dos métodos básicos necessários para integrar com cada uma das administradoras realizou-se um detalhamento de cada um dos serviços listados no quadro 17, sendo que para melhor compreensão este detalhamento está anexado junto ao apêndice A e B do corrente trabalho.

Com a elicitação dos campos e de cada método/serviço bastou realizar uma solicitação as equipes de suporte de cada administradora de PEF, a fins de obter os dados de autenticação em ambiente de homologação para posterior comunicação via Web Service.

Para finalizar os passos da elicitação de requisitos definiu-se o ambiente que o aplicativo iria ser executado (web ou *desktop*), sendo que optou-se por web

após realizadas entrevistas com programadores DataFlex.

Para realização total do levantamento descrito nesta seção utilizou-se conhecimentos adquiridos em experiências próprias na área de negócios voltados para transporte rodoviário de cargas. As ferramentas utilizadas na presente seção são Microsoft Outlook e Word.

Visto que a execução da elicitação de requisitos não é o bastante para implementação da integração com mais de uma administradora, houve a necessidade da realização da próxima etapa que apresenta a modelagem lógica das tabelas necessárias para prototipação e posterior implementação do trabalho.

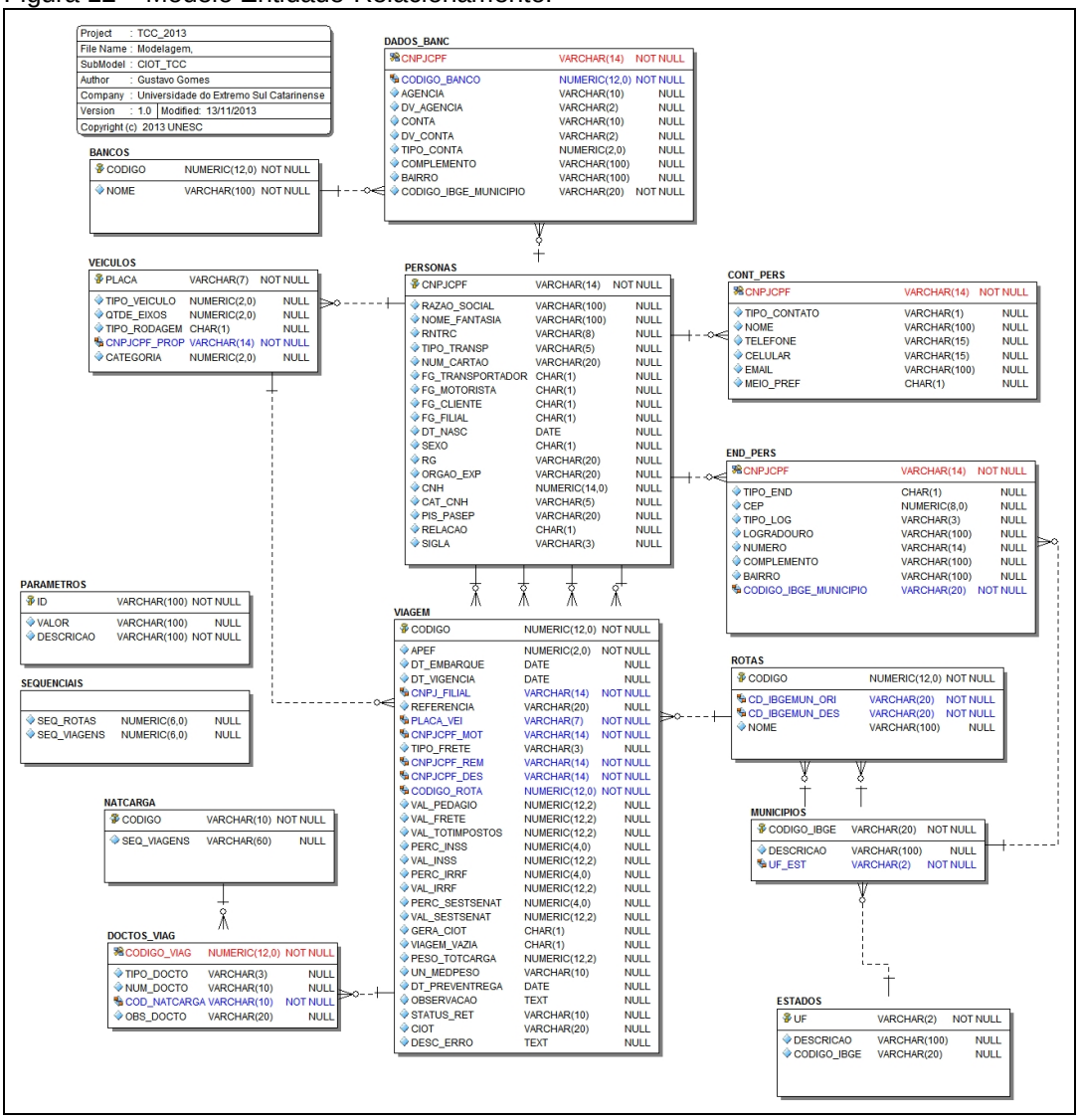
7.1.3 Modelagem Lógica

A criação da modelagem lógica do sistema é essencial para que seja criada uma ferramenta de comunicação padrão, com banco de dados e aplicação, ambos sem redundâncias, desta forma é possível atender a integração com no mínimo duas administradoras PEF utilizando o mesmo banco de dados e também facilita o conhecimento a cerca da ferramenta em futuras manutenções e novas implementações.

No presente trabalho a modelagem foi criada de forma sucinta, evitando pleonasmos e facilitando a interpretação do aplicativo implementado. As tabelas, campos, definição de tipos de campos, verificação de *primary key* e *foreign key* foram modeladas com base no elicitação obtida na seção 7.1.2 e visando a próxima etapa de prototipação e desenvolvimento.

O modelo lógico final é composto por 14 tabelas, representado pelo modelo Entidade-Relacionamento a seguir (figura 12), que por sua vez foi implementado no *software* ER/Studio.

Figura 12 – Modelo Entidade-Relacionamento.



Fonte: Do autor.

Logo abaixo, no quadro 18 será disposto o dicionário de dados com a principal tabela do processo de integração com as administradoras de pagamento eletrônico de frete, a tabela VIAGEM. A definição de outras tabelas mostradas na modelagem na figura 12 estão dispostas do apêndice C do corrente trabalho.

Quadro 18 – Dicionário de dados da tabela de viagens.

Viagem – Tabela que contém informações da viagem que será integrada a administradora			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO	NUMERIC(12)	Código interno da viagem, único
FK	PLACA_VEI	VARCHAR(7)	Placa do veículo, formato XXX9999
FK	CNPJ_FILIAL	VARCHAR(14)	CNPJ da filial, sem formatação
FK	CNPJCPF_MOT	VARCHAR(14)	CPF do motorista, sem formatação
FK	CNPJCPF_REM	VARCHAR(14)	CNPJ/CPF do cliente remetente, sem formatação
FK	CNPJCPF_DES	VARCHAR(14)	CNPJ/CPF do cliente destinatário, sem formatação
FK	CODIGO_ROTA	NUMERIC(12)	Código interno da rota
	APEF	NUMERIC(2)	Código que representa a administradora de pagamento eletrônico de frete, possível selecionar: 1 = APISUL 2 = DBTRANS
	DT_EMBARQUE	DATE	Data do embarque para início do transporte
	DT_VIGENCIA	DATE	Data da vigência da viagem
	REFERENCIA	VARCHAR(20)	Referência da viagem para a administradora
	TIPO_FRETE	VARCHAR(3)	Tipo de frete, possível selecionar: CIF FOB
	VAL_PEDAGIO	NUMERIC(12,2)	Valor do pedágio a pagar na rota
	VAL_FRETE	NUMERIC(12,2)	Valor do frete a pagar sem impostos somados
	VAL_TOTIMPOSTOS	NUMERIC(12,2)	Valor total de impostos retidos
	PERC_INSS	NUMERIC 4	Percentual de INSS retido
	VAL_INSS	NUMERIC(12,2)	Valor de INSS retido
	PERC_IRRF	NUMERIC 4	Percentual de IRRF retido
	VAL_IRRF	NUMERIC(12,2)	Valor de IRRF retido
	PERC_SESTSENAT	NUMERIC 4	Percentual de SEST/SENAT retido
	VAL_SESTSENAT	NUMERIC(12,2)	Valor de SEST/SENAT retido
	GERA_CIOT	CHAR(1)	Gera CIOT? Possível selecionar: S = SIM N = NÃO
	VIAGEM_VAZIA	CHAR(1)	A viagem é sem carga? Possível selecionar: S = SIM N = NÃO
	PESO_TOTCARGA	NUMERIC(12,2)	Peso total da carga transportada
	UN_MEDPESO	VARCHAR(10)	Unidade de medida do peso
	DT_PREVENTREGA	DATE	Data de previsão de entrega da carga transportada
	OBSERVAÇÃO	TEXT	Observação livre
	STATUS_RET	VARCHAR(10)	Status de retorno da comunicação com a administradora, possíveis retornos: SUCESSO ERRO
	CIOT	VARCHAR(20)	Código Identificador da Operação de Transporte
	DESC_ERRO	TEXT	Descrição do erro retornado pela administradora quando o STATUS_RET = ERRO

Fonte: Do autor.

Os campos criados na tabela mostrada no quadro 18 são idôneos para no mínimo atender a resolução 3.658/11 utilizando a administradora Dbtrans ou Apisul, independente da administradora de PEF que está sendo utilizada, porém para a percepção concreta deste padrão foram criadas as etapas de prototipação e por seqüente a implementação.

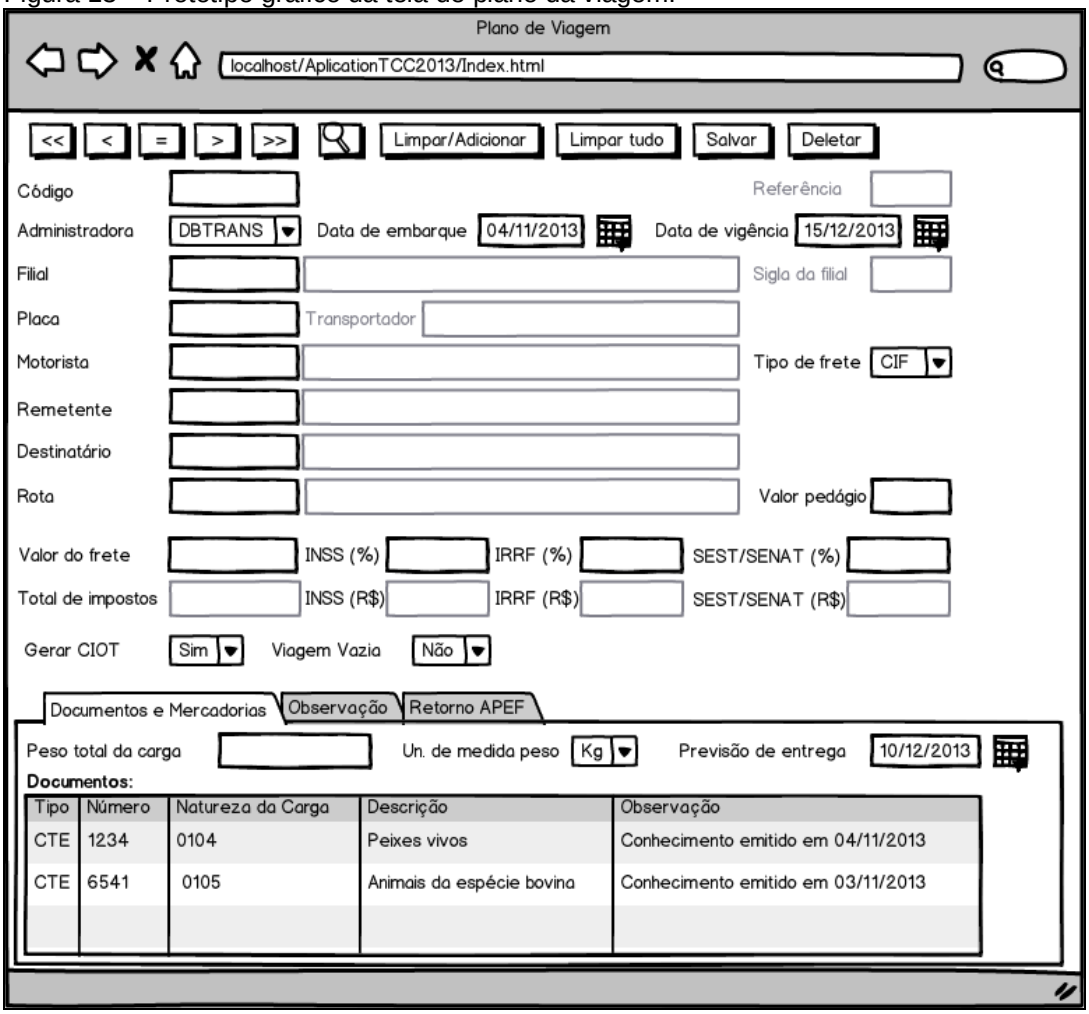
7.1.4 Prototipação

A criação de telas de protótipo é a última etapa antes de produção da ferramenta abordada no trabalho. Nesta fase não necessariamente será representada todas suas funcionalidades e a interface real da aplicação, mas somente um exemplar sem funções inteligentes, com esboço gráfico para fins de ilustração e entendimento do que se planeja desenvolver.

Assim como realizou-se na seção de modelagem lógica, a presente seção apresentará o protótipo da principal janela da ferramenta, utilizando o aplicativo *Balsamiq Mockups*. As demais telas do sistema também foram prototipadas e estão sendo mencionadas no apêndice D do trabalho atual.

O design gráfico da tela de plano de viagem prototipada está sendo mostrada na figura 13.

Figura 13 – Protótipo gráfico da tela de plano da viagem.



Fonte: Do autor.

Após apresentação do design gráfico da tela mostrada na figura 13 e análise dos campos da tela conforme levantamento dos requisitos, serão listados no quadro 19 uma breve descrição de cada componente da tela, com algumas regras de interface.

Quadro 19 – Regras de interface

Componente	Regra/descrição
Código	Auto-incremento. Código utilizado para consultar um plano de viagem.
Referência	Inacessível. Mostrar neste campo a sigla da filial concatenada ao código da viagem.
Administradora	DBTRANS é a opção sugerida. As opções existentes na combo são DBTRANS e APISUL.
Data de embarque	A data deve ser maior ou igual à data atual.
Data de vigência	A data deve ser maior ou igual à data atual e de embarque.
Filial	Após informar o CNPJ da filial os campos de nome e sigla da filial devem ser mantidos inacessíveis.
Placa	Após informar a placa o campo de transportador deve ser mantido inacessível.
Motorista	Após informar um CPF de motorista o campo de nome deve ser mantido inacessível.
Tipo de frete	CIF é a opção sugerida. As opções existentes na combo são CIF (pagador do frete é o remetente) e FOB (pagador do frete é o destinatário).
Remetente	Após informar o CNPJ do cliente remetente o campo de nome deve ser mantido inacessível.
Destinatário	Após informar o CNPJ do cliente destinatário o campo de nome deve ser mantido inacessível.
Rota	Após informar o código da rota o campo de nome de ver mantido inacessível.
Valor do frete	O valor do frete deve ser maior que zero.
INSS (%)	O percentual de INSS retido deve ser maior que zero.
IRRF (%)	O percentual de IRRF retido deve ser maior que zero.
SEST/SENAT (%)	O percentual de SEST/SENAT retido deve ser maior que zero.
Total de impostos	Este campo sempre estará inacessível e mostrará automaticamente o valor total de impostos conforme valores mostrados nos campos de INSS (R$), IRRF (R$) e SEST/SENAT (R$).
INSS (R$)	Este campo sempre estará inacessível e mostrará o valor do INSS retido de acordo com o valor do frete e do percentual de imposto informado.
IRRF (R$)	Este campo sempre estará inacessível e mostrará o valor do IRRF retido de acordo com o valor do frete e do percentual de imposto informado.
SEST/SENAT (R$)	Este campo sempre estará inacessível e mostrará o valor do SEST/SENAT retido de acordo com o valor do frete e do percentual de imposto informado.
Gerar CIOT	Sim é a opção sugerida. As opções existentes na combo são Sim e Não.
Viagem Vazia	Não é a opção sugerida. As opções existentes na combo são Sim e Não.
Documentos e Mercadorias	Nesta aba devem ser preenchidos os documentos que comprovam o transporte da carga, bem como o peso, a unidade de medida do peso e a previsão de entrega das mercadorias. A previsão de entrega deve ser maior ou igual a data atual.
Observação	Na aba de Observação poderá ser informada uma observação qualquer para viagem.
Retorno APEF	Na aba APEF serão mostrados os dados de retorno logo após que for salvo o plano de viagem e a comunicação ser realizada com a administradora. Os dados devem ser mostrados de forma inacessível, sendo que poderão ser visualizados nesta aba os campos de status, CIOT e descrição do erro retornado.

Fonte: Do autor.

As regras de interface descritas no quadro 19 podem evitar a falha de comunicação no momento que a administradora de pagamento eletrônico de frete retornar uma requisição, ou seja, quando há falha de comunicação as administradoras simplesmente retornam a mensagem vazia. As demais regras para geração do CIOT e conseqüente cumprimento da resolução 3.658/11 ficam a cargo de cada

administradora, não sendo necessário aplicar as mesmas na ferramenta abordada no presente trabalho.

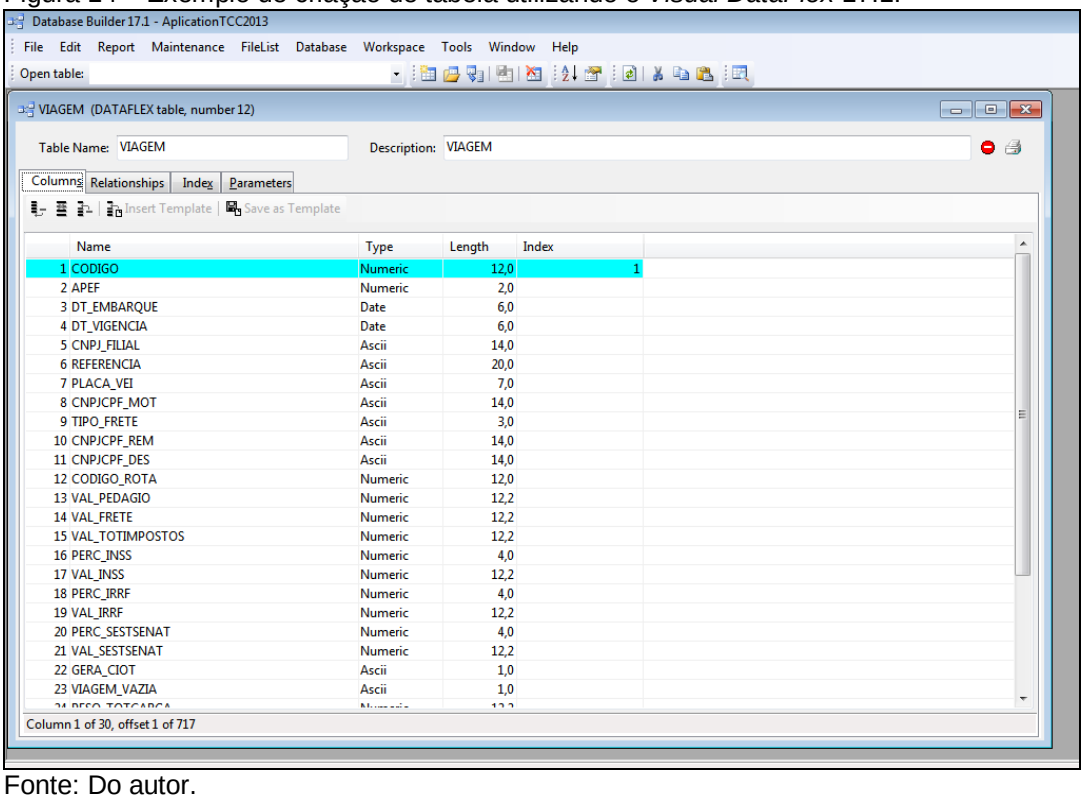
Com a prototipação, regras de interface, e demais etapas citadas nas seções anteriores da metodologia concluídas, a próxima seção abordará a implementação da ferramenta.

7.1.5 Implementação

A implementação é a última etapa da ferramenta abordada neste trabalho, onde se aplica todas as outras principais etapas citadas neste capítulo, sendo os seguintes: levantamento de requisitos, modelagem lógica e prototipação. Ambas as etapas do capítulo atual pertencem a áreas da engenharia de software.

Esta fase de desenvolvimento do trabalho iniciou-se com a criação do banco, utilizando o próprio banco nativo e disponibilizado pelo *Visual DataFlex* representado pelo exemplo figura 14.

Figura 14 – Exemplo de criação de tabela utilizando o *Visual DataFlex* 17.1.

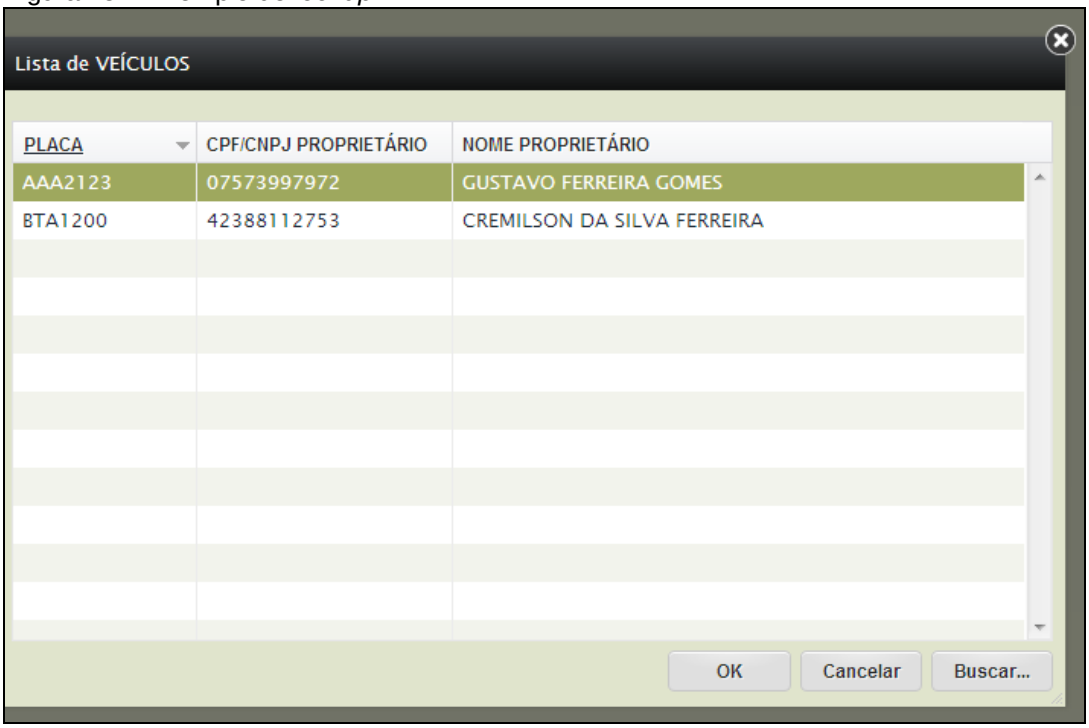


Fonte: Do autor.

Com o banco de dados nativo criado no *Database Builder* do *Visual DataFlex* foram criados respectivamente o dicionário de dados de cada tabela

utilizada e foram desenvolvidas as telas de consulta denominadas na IDE como *lookups* (pesquisas), conforme visualizado na figura 15.

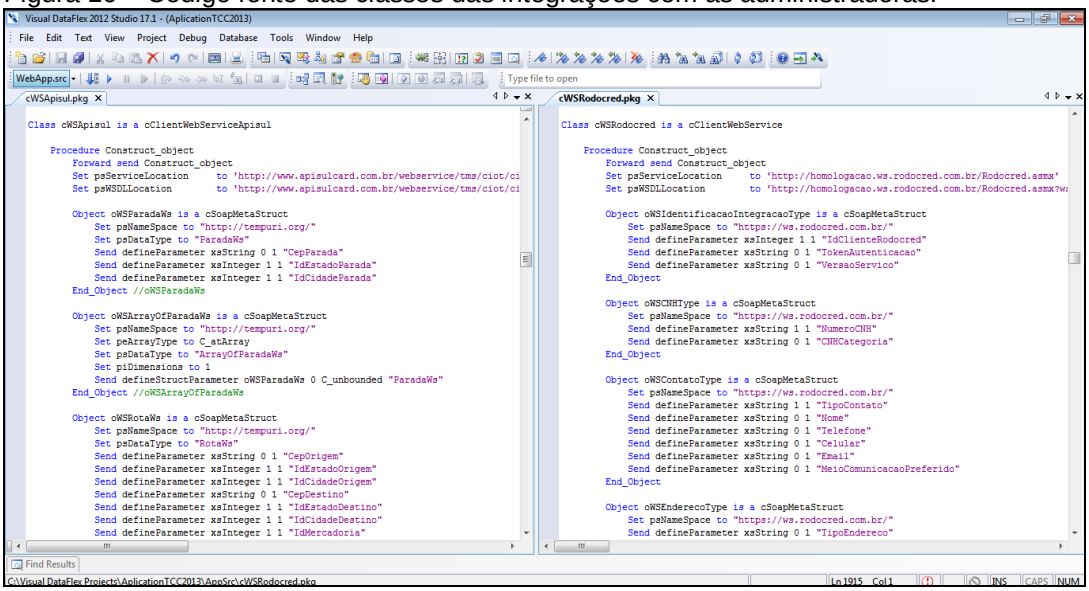
Figura 15 – Exemplo de *lookup*.



Fonte: Do autor.

Posteriormente a criação da base para desenvolvimento mencionada acima (criação de tabelas, dicionário de dados e consultas), iniciou-se o processo de integração via Web Service com as administradoras (conforme figura 16) utilizando a mesma estrutura de banco de dados e compartilhando objetos em comum em uma única função.

Figura 16 – Código fonte das classes das integrações com as administradoras.



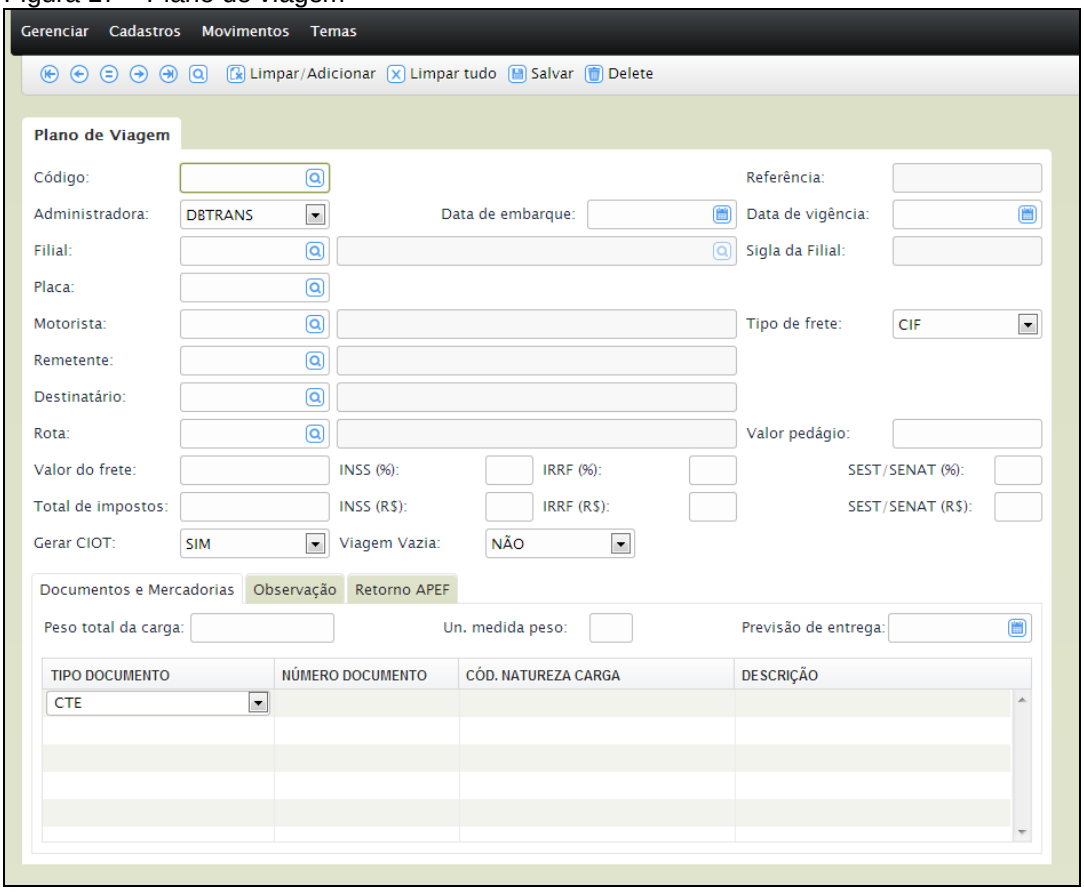
Fonte: Do autor.

O desenvolvimento dos pacotes de comunicação foram implementados consumindo o Web Service de homologação de ambas as administradoras a partir do WSDL disponibilizado em cada manual, da Apisul e Dbtrans.

Para permitir que o usuário final tenha acesso ao padrão de integração foram desenvolvidas as telas de cadastros e também a principal tela já mencionada na prototipação, a de plano de viagem. Para chamada de cada tela foi criado um menu disposto na barra superior de modo horizontal.

Esta tela principal denominou-se plano de viagem e pode ser visualizada na figura 17.

Figura 17 – Plano de viagem



Fonte: Do autor.

Ao utilizar a tela mencionada na figura 17, é possível criar uma viagem apontando qual administradora será realizada para que a resolução 3.658/11 da ANTT seja atendida.

Por fim, a resolução vigente é atendida quando o Web Service da administradora de PEF escolhida para o plano de viagem corrente retorna o status de sucesso e um número de CIOT, onde ambos os campos estão dispostos na aba de “Retorno APEF” desta tela.

7.2 RESULTADOS OBTIDOS

Aplicando e realizando a junção do conhecimento adquirido no decorrer deste trabalho de conclusão de curso, e em toda a graduação do curso de Ciência da Computação e também por alguns anos de experiência na profissão de analista de sistemas e negócios na área de transporte rodoviário de cargas foi possível obter os resultados esperados. As etapas descritas neste capítulo demonstram de uma forma concreta que os objetivos do trabalho proposto foram realmente atingidos.

Através do levantamento bibliográfico e elicitação dos requisitos, foi possível definir o que é necessário para criação de um aplicativo que atendesse a resolução 3.658/11 da ANTT independente das particularidades de duas administradoras de pagamento eletrônico de frete.

Utilizando técnicas e modelos da engenharia de *software* foi possível criar a ferramenta de integração com telas organizadas e limpas, com poucas tabelas se comparado a quantidade de serviços que as administradoras disponibilizam, sendo que os serviços que não foram utilizados, não são relevantes perante a legislação apontada neste trabalho. Com isso, comprovou-se que é possível reduzir o tempo com estes tipos de integrações, que são gastos por equipes de TI das transportadoras ou terceirizadas.

Na realização de testes de comunicação com ambas as administradoras utilizando a tela de plano de viagem, possibilitou-se consumir o número do CIOT, que o código que representa que a resolução 3.658/11 foi atendida com sucesso.

8 CONCLUSÃO

Atualmente, um dos grandes fatores de insatisfação por parte dos clientes que adquirem serviços ou produtos das empresas de desenvolvimento de software é a falta de qualidade, fazendo com que as atualizações e/ou instalações dos softwares sejam liberados ao cliente com inconsistência dificultando seu trabalho e causando atrasos e prejuízos financeiros. Outro fator não menos importante referente à insatisfação dos clientes na área de TI, é a baixa usabilidade dos sistemas que por sua vez é proveniente de um levantamento de requisitos inexistente ou falho.

A elicitação de requisitos não é uma tarefa fácil de ser executada, pois exige um conhecimento aprofundado sobre técnicas de engenharia e manipulação de informações, onde as mesmas, muitas vezes são mal interpretadas ou ignoradas pelo analista, gerando por algumas vezes falhas no sistema.

Com este trabalho, foi possível contribuir para a evolução das técnicas de elicitação de requisitos e modelagem lógica de dados, de uma forma estruturada e seqüencial, aumentando o próprio acumulo de conhecimento no que diz respeito à Engenharia de *Software* e Engenharia de requisitos.

Visando evitar a insatisfação dos clientes e a dissipação de tempo de desenvolvimento, implementou-se neste trabalho uma ferramenta única de comunicação via Web Service com mais de uma administradora de Pagamento Eletrônico de Frete.

Para alcançar o sucesso no desenvolvimento da ferramenta foi realizada a elicitação de requisitos de forma documental, utilizando o próprio Word, da Microsoft. Também foram utilizadas as ferramentas ER/Studio e *Balsamiq Mockups* para a modelagem e prototipação do aplicativo, e por fim para a implementação final foi utilizado o Visual *DataFlex*, da empresa Data Acess.

Embora a maior dificuldade tenha sido lhe dar com inúmeras atualizações que ocorreram mensalmente no Web Service e na documentação disponibilizada pelas administradoras no decorrer do desenvolvimento deste trabalho, foi possível através da contribuição do levantamento de requisitos realizado de forma técnica e minuciosa, atender de forma limpa e simples a resolução disposta pela Agência Nacional de Transportes Terrestres no ano de 2011, que é a resolução 3.658/11.

Pode-se concluir que, a demora pelo atendimento desta resolução por parte das empresas desenvolvedoras de sistemas da área de transporte tem sido um

dos principais motivos, na qual muitas empresas de transporte rodoviário de carga não se satisfazem com o sistema que possui, desta forma pode-se concluir que a solução apresentada neste trabalho para o atendimento da resolução contribui direta e indiretamente com as boas praticas de engenharia de software e de requisitos e com a satisfação dos transportadores.

REFERÊNCIAS

ANTT. **Empresas habilitadas como Administradoras de Meios de Pagamento Eletrônico.** Disponível em: <http://www.antt.gov.br/index.php/content/view/12674/Empresas_habilitadas_como_Administradoras_de_Meios_de_Pagamento_Eletronico.html>. Acesso em: 10 abr. 2013.

BANKI, André Luis; TANAKA, Sérgio Akio. Metodologias ágeis: uma visão prática. **Engenharia de *Software* Magazine**, Rio de Janeiro, ano 1, ed. 4, p. 22-29, 2008.

BEZERRA, Eduardo. **Princípios de analise e projeto de sistemas com UML.** Rio de Janeiro: Campus, 2003. 286 p. Disponível em: <http://wiki.sj.ifsc.edu.br/wiki/index.php/Ciclo_de_Vida_Iterativo_e_Incremental>. Acesso em: 25 mai. 2013.

BRASIL. Resolução nº 3.658, de 19 de Abril de 2011. **Diário Oficial [da] República Federativa do Brasil**, Poder Executivo, Brasília, DF, 27 abr. 2011. Seção 1, p. 97.

CZERVENY, Arno. **Web Services: Onde estamos e para onde vamos.** Revista Mundo Java, Ano I nº 02, Mundo OO, Rio de Janeiro, RJ. 2004.

DBTRANS. **DETALHAMENTO TÉCNICO DA INTEGRAÇÃO RODOCRED:** Versão 2.2. Rio de Janeiro: Dbtrans, 2011. 118 p.

DBTRANS. **DBTRANS é habilitada pela ANTT para pagamento eletrônico de frete.** Disponível em: <http://www.rodocred.com.br/Imprensa/Releases/DBTRANS%C3%A9habilitadaparapagamentoeletr%C3%B4nico/tabid/518/Default.aspx>. Acesso em: 10 abr. 2013.

FALBO, Ricardo de Almeida. **Engenharia de Requisitos.** Espírito Santo, 2011. UFES-Universidade Federal do Espírito Santo. Disponível em: <www.inf.ufes.br/~falbo/files/Notas_Aula_Engenharia_Requisitos.pdf >. Acesso em: 22 mai. 2013.

FERRARI, Thales Martins. **Sistema de *Sofware* para auxílio dos docentes da Unifei**. 2006. 138 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal de Itajubá, Minas Gerais.

FOWLER, Martin. **A nova metodologia.** 2003. Disponível em: <http://www.martinfowler.com/articles/newMethodology.html>. Acesso em: 25 mai. 2013.

GOMIDE, Fernando. **Fundamentos de modelagem de sistemas.** 2007. DCA-FEECUnicamp. Curso IA 881 Otimização Linear da Faculdade de Engenharia Elétrica e de Computação. Disponível em: <http://www.dca.fee.unicamp.br/~gomide/courses/IA881/transp/IA881Modelagem.pdf> . Acesso em : 27 mai. 2013.

JUNQUEIRA, Fabrício. **Modelagem e simulação distribuídas de sistemas produtivos.** 2006. 222 f. Tese (Doutorado em Engenharia Mecatrônica e de Sistemas Mecânicos). Escola Politécnica da Universidade de São Paulo, São Paulo.

LESSA, Rafael Orivaldo; LESSA JUNIOR, Edson Orivaldo. **Modelos de processos de engenharia de *software*.** Palhoça, 2010. Disponível em: <http://ebookbrowse.com/02-artigo-pdf-d25204912 >. Acesso em: 24 mai. 2013.

LUDVIG, Diogo. REINERT, Jonatas Davson. **Estudo do uso de Metodologias Ágeis no Desenvolvimento de uma Aplicação de Governo Eletrônico.** 2007. Departamento de Informática e Estatística – Universidade Federal de Santa Catarina Disponível em: <http://projetos.inf.ufsc.br/arquivos_projetos/projeto_589/Artigo_Diogo_Jonatas.pdf>. Acesso em: 26 mai. 2013.

MAZZOLA, Vitório Bruno. **Engenharia de *Software*:** Conceitos Básicos. Florianópolis, 2010. INE/CTC/UFSC. Disponível em: <http://algol.dcc.ufla.br/~monserrat/icc/Introducao_ES.pdf >. Acesso em 19 mai. 2013.

OLIVEIRA, Luiz Ângelo. **Ciclo de vida de sistemas de informação.** São Paulo, 2009. Disponível em: < http://www.eteavare.com.br/arquivos/43_37.pdf >. Acesso em: 22 mai. 2013.

PAULON, Renan. **Web Services in JAVA.** Disponível em: <http://www.imasters.com.br/artigo/1863>. Acesso em: 30 mai. 2013.

PARREIRA JUNIOR, Walteno Martins. **Engenharia de *Software*.** Universidade do Estado de Minas Gerais. 2011. 108 f. Apostila - Fundação Educacional de Ituiutaba. Curso de Engenharia da Computação e Engenharia de *Software*. Disponível em: <http://www.waltenomartins.com.br/ap_es_v1.pdf>. Acesso em: 28 abr. 2013.

PORTELLA, Cristiano R. R.. **Análise de Requisitos - Conceitos**. Disponível em: <http://www.paiossin.com/wordpress/wp-content/uploads/2011/11/Anlise-de-Requisitos-Conceitos.pdf>. Acesso em: 10 out. 2013.

POTTS, Stephen; KOPACK, Mike. **Aprenda Web services em 24 horas**, tradução de Marcos Vieira. Rio de Janeiro: Campus, 2003. 367 p.

PRESSMAN, Roger S. **Engenharia de *Software*.** 6. ed. São Paulo: McGraw-Hill, 2006.

SATO, Danilo Toshiaki. **Uso eficaz de métricas em métodos ágeis de desenvolvimento de *software*.** 2007. 139 f. Dissertação (Mestrado em Ciências da Computação) – Universidade de São Paulo, São Paulo.

SAVI, Antonio Francisco. **Modelo de sistema para gerenciamento de conhecimentos explícitos em abordagens de DFA (Design For Assembly).** Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2009.

SILVA, Cristiano Campos; SOUZA, Kleyton Farias de; DANTAS, Samuel Dias. **Metodes: Metodologia de desenvolvimento de *software*.** 2006. 159 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Faculdade Cenecista de Brasília. Distrito Federal.

SILVA, Luís César. **Simulação de Processos.** Universidade Federal do Espírito Santo. 2005. Disponível em: < http://www.agais.com/simula.htm>. Acesso em: 26 mai. 2013.

SOARES, Michel dos Santos. **Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de *Software*.** DCC/UFLA – RCC/Infocomp. Vol. 3, n. 2. Minas Gerais, nov. 2004. Artigos. Disponível em: <http://www.dcc.ufla.br/infocomp/artigos/v3.2/art02.pdf>. Acesso em: 28 abr. 2013.

SOMMERVILLE, Ian. **Engenharia de *Software*.** São Paulo: Addison-Wesley, 2003. 592 p.

APÊNDICE(S)

APÊNDICE A – Quadros de elicitação de requisitos da administradora de PEF Dbtrans

Método Autenticar Cliente		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
iIdClienteRodocred	Identificação do cliente Rodocred	N
sLoginIntegracao	Login do Usuário de Integração	S
SChaveAutenticacao	Senha de Autenticação	S

Método Manter Transportador		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
CNPJCPFTransportador	CNPJ / CNPF do transportador	S
RNTRC	RNTRC do transportador	S
NomeTransportador	Nome do transportador	S
TipoTransportador	Tipo do transportador: TAC, ETC ou CTC	S
NumeroCartao	Número do cartão do transportador	N
DataNascimento	Data de nascimento pessoa física	D
Identidade	RG pessoa física	S
OrgaoExpedidor	Orgão expedidor pessoa física	S
PISPASEP	PIS/PASEP pessoa física	N
Sexo	Sexo pessoa física	S
Contato.TipoContato	Tipo de contato	S
Contato.Nome	Nome para contato	S
Contato.Telefone	Telefone para contato	S
Contato.Celular	Celular para contato	S
Contato.Email	Email para contato	S
Contato.MeioComunicacaoPreferido	Meio de comunicação preferido	S
Endereco.TipoEndereco	Tipo de endereço: Comercial, entrega ou residencial	S
Endereco.CEP	Cep	S
Endereco.TipoLogradouro	Tipo de logradouro	S
Endereco.Logradouro	Logradouro	S
Endereco.Numero	Número	S
Endereco.Complemento	Complemento	S
Endereco.Bairro	Bairro	S
Endereco.Cidade	Cidade	S
Endereco.Estado	Estado	S

Método Manter Veículo		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
CNPJCPFTransportador	CNPJ / CNPF do transportador	S
RNTRC	RNTRC do transportador	S
NomeTransportador	Nome do transportador	S
TipoTransporteVeiculo	Tipo de contrato vinculado a veículo: Terceiro, Frota ou Agregado	S
TipoVeiculo	Tipo de veículo	S
QtdeEixos	Quantidade de ixos	N
TipoRodagem	Tipo de rodagem	S
PlacaVeiculo	Placa do veículo	S
UfPlacaVeiculo	UF placa veículo	S
NumeroFrotaVeiculo	Número da frota do veículo	N
TipoCombustivelVeiculo	Tipo de combustível	S
CapacidadeTanqueVeiculo	Capacidade do tanque em litros	N
MediaConsumoVeiculo	Média de consumo do veículo	N

Método Consultar Rotas		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
Nome Rota	Nome da rota no sistema Rodocred (Dbtrans)	S

Método Manter Viagem		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
IdClienteRodocred	Identificação do Cliente Rodocred	N
TokenAutenticacao	Token de autenticação	S
VersaoServico	Versão Serviço	S
Operação	Tipo de Operação a ser realizada	S
TipoViagem	Tipo de viagem	S
Embarque	Data e hora de embarque da viagem	D
PrevisaoEntrega	Data e hora de previsão de entrega da viagem	D
Foro.Estado	UF do foro do contrato de frete	N
Foro.Cidade	Cidade do logradouro do endereço	N
DocumentoRef	Número de documento de identificação da viagem para o contratante	S
DataVigencia	Data e hora da vigência da viagem	D
Observação	Indicador de geração de CIOT.	S
SemTransporteCarga	Indicador de Viagens sem Transporte de Carga	S
Transportador[]	Dados do transportador da viagem	-
RotaViagem[]	Rota definida para a viagem	-
VeiculosViagem[]	Veículos relacionados a viagem	-
MotoristaViagem[]	Dados dos motoristas na viagem	-
EnvolvidosTransporteViagem[]	Dados dos envolvidos do transporte na viagem: Remetente, Destinatário e Consignatário	-
ValoresViagem	Valores financeiros relacionados a viagem	N
ImpostosViagem[]	Valores financeiros de impostos da viagem	-
DocumentosViagem[]	Lista de Documentos Relacionados a uma determinada Viagem	-
ProgramacaoViagem[]	Armazena as informações das operações da viagem	-

APÊNDICE B – Quadro de elicitação de requisitos da administradora de PEF Apisul

Método Declaração Operação Transporte		
Campos obrigatórios para integrar	**Descrição**	**Tipo** **S=String** **N = Numeric** **D = Date**
CodigoTransporadora	Código da transportadora	S
Viagem.Id	Id da viagem	N
Viagem.Tipo	Tipo da viagem	S
Viagem.DataInicio	Data de inicio da viagem	D
Viagem.DataFim	Data final da viagem	D
Viagem.idTipoRota	Tipo de rota	N
Viagem.idCategoriaVeiculo	Categoria do veículo	N
Viagem.ValorFrete	Valor do frete	N
Viagem.CalcularPedagio	Calcular Pedagio	S
Viagem.ValorPedagio	Valor do pedágio	N
Viagem.ValorSestSenat	Valor do Sest/Senat	N
Viagem.ValorIrrf	Valor do Irrf	N
Viagem.ValorInss	Valor do Inss	N
Viagem.ValorImpostos	Valor total de impostos	N
Rota[]	Rota da viagem	-
Destinatario[]	Destinatário da viagem	-
Contratante[]	Contratante da viagem	-
Contratado[]	Contratad da viagem	-
Veiculos[]	Veículos envolvidos	-
Favorecido[]	Dados bancários do favorecido da viagem	-

APÊNDICE C – Quadros dos dicionários de dados das tabelas

NatCarga – Tabela que contém naturezas da carga transportada			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO	VARCHAR(10)	Código da natureza da carga
	DESCRICAO	VARCHAR(60)	Descrição da natureza da carga

Doctos_Viag – Tabela que contém os documentos da viagem			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO_VIAG	NUMERIC(12)	Código da viagem
	TIPO_DOCTO	VARCHAR(3)	Tipo de documento
	NUM_DOCTO	VARCHAR(10)	Número do documento
FK	COD_NATCARG	VARCHAR(10)	Código da natureza da carga
	OBS_DOCTO	VARCHAR(20)	Observação do documento

Bancos – Tabela que contém cadastrados os bancos			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO	NUMERIC(12)	Código do banco conforme FEBRABAN
	NOME	VARCHAR(100)	Nome do banco

Estados – Tabela que contém cadastrados os estados			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	UF	VARCHAR(2)	UF
	DESCRICAO	VARCHAR(100)	Descrição
	CODIGO_IBGE	VARCHAR(20)	Código do IBGE

Municipios – Tabela que contém cadastrados os municípios			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO_IBGE	VARCHAR(20)	Código IBGE do município
	DESCRICAO	VARCHAR(100)	Descrição do município
FK	UF_EST	VARCHAR(2)	UF do estado

Rotas – Tabela que contém cadastradas as rotas			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CODIGO	NUMERIC(12)	Código interno da rota
	NOME	VARCHAR(100)	Nome da rota
FK	CD_IBGEMUN_ORI	VARCHAR(20)	Código IBGE do município de origem da rota
FK	CD_IBGEMUN_DES	VARCHAR(20)	Código IBGE do município de destino da rota

Parametros – Tabela que contém os parâmetros do sistema			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	ID	VARCHAR(100)	ID que representa o parâmetro
	VALOR	VARCHAR(100)	Valor do parâmetro
	DESCRICAO	VARCHAR(100)	Descrição do parâmetro

Sequenciais – Tabela do sistema de controle dos sequenciais			
Chave	**Atributo**	**Tipo**	**Comentário**
	SEQ_ROTAS	VARCHAR(100)	Sequencial de rotas
	SEQ_VIAGENS	VARCHAR(100)	Sequencial de viagens

Personas – Tabela que contém os dados dos Personas (filial, transportador, cliente, motorista)			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CNPJ_CPJ	VARCHAR(14)	CNPJ/CPF dos PERSONAS
	RAZAO_SOCIAL	VARCHAR(100)	Razão Social/Nome
	NOME_FANTASIA	VARCHAR(100)	Nome fantasia
	RNTRC	VARCHAR(8)	RNTRC
	TIPO_TRANSP	VARCHAR(5)	Tipo transportador
	NUM_CARTAO	VARCHAR(20)	Número do cartão
	FG_TRANSPORTADOR	CHAR(1)	Identificador de Transportador
	FG_MOTORISTA	CHAR(1)	Identificador de Motorista
	FG_CLIENTE	CHAR(1)	Identificador de Cliente
	FG_FILIAL	CHAR(1)	Identificador de Filial
	DT_NASC	DATE	Data de nascimento
	SEXO	CHAR(1)	Sexo
	RG	VARCHAR(20)	Identidade
	ORGAO_EXP	VARCHAR(20)	Orgão expedidor RG
	CNH	NUMERIC(14)	CNH
	CAT_CNH	VARCHAR(5)	Categoria CNH
	PIS_PASEP	VARCHAR(20)	Núm. PIS/PASEP
	RELACAO	CHAR(1)	Relação do transp. Com a empresa
	SIGLA	VARCHAR(3)	Sigla da filial

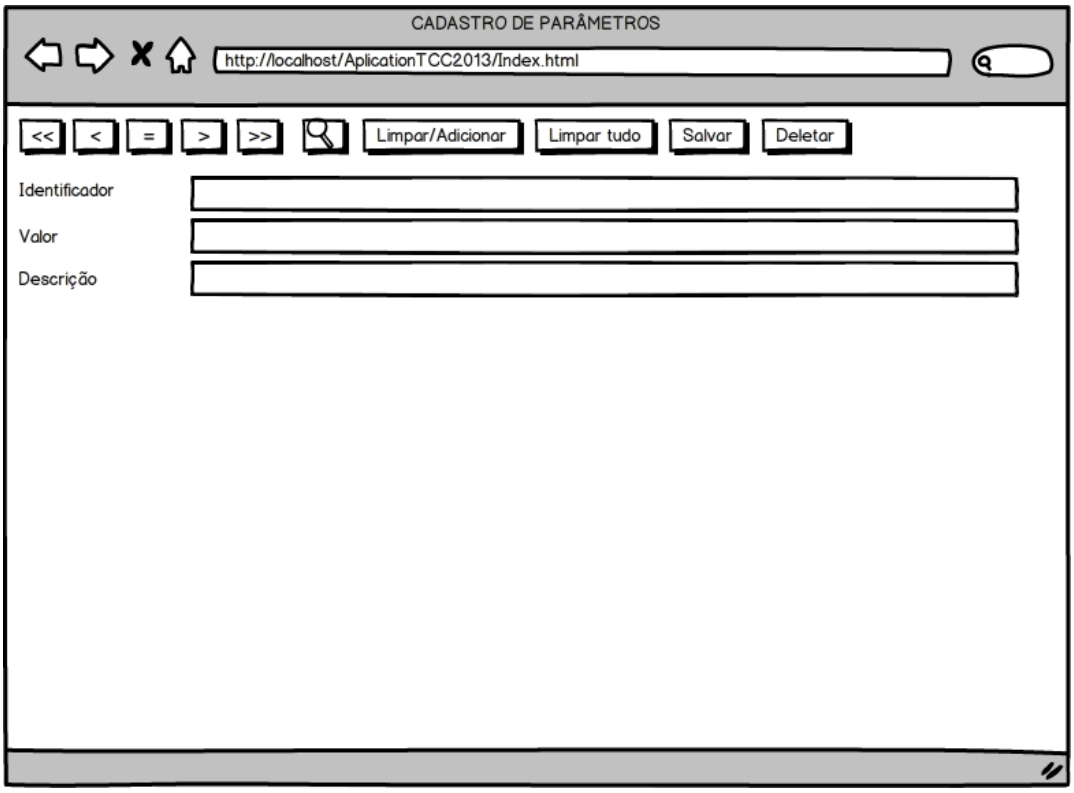
Cont_Pers – Tabela que contém os contatos dos Personas			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CNPJ_CPJ	VARCHAR(14)	CNPJ/CPF do PERSONAS
	TIPO_CONTATO	VARCHAR(1)	Tipo de contato
	NOME	VARCHAR(100)	Nome
	TELEFONE	VARCHAR(15)	Telefone
	CELULAR	VARCHAR(15)	Celular
	EMAIL	VARCHAR(100)	Email
	MEIO_PREF	CHAR(1)	Meio para comunicação preferido

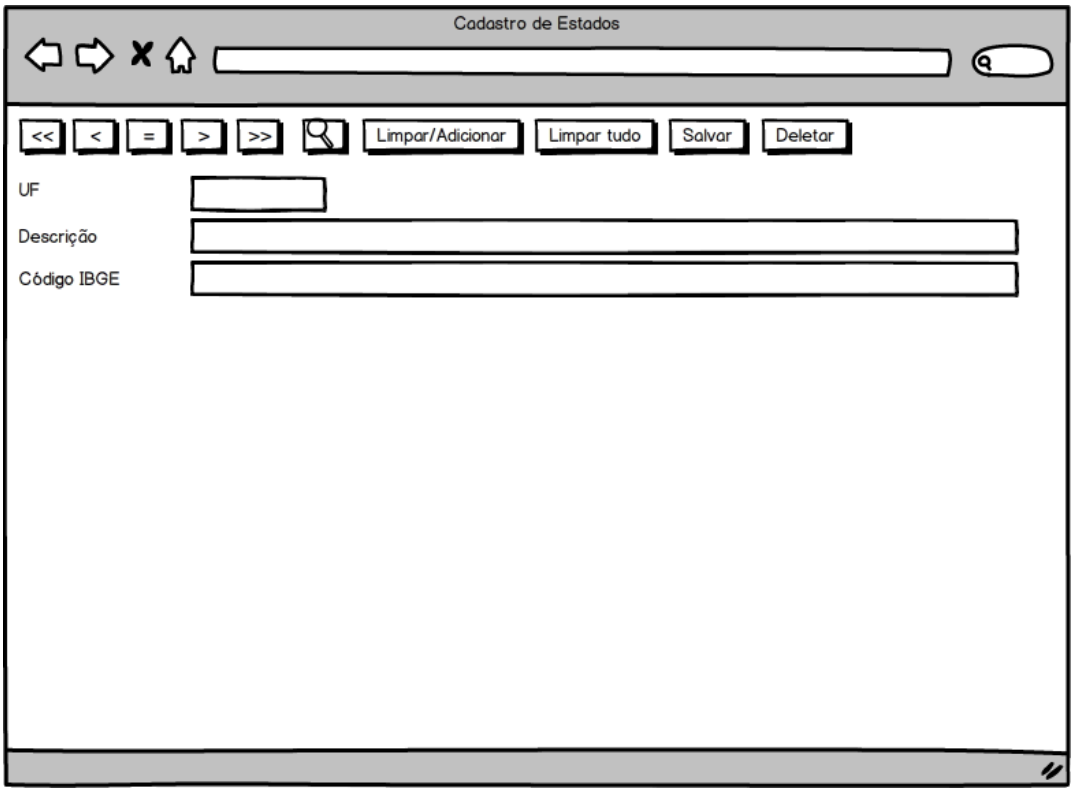
Dados_Banc – Tabela que contém os dados bancários dos Personas			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CNPJ_CPJ	VARCHAR(14)	CNPJ/CPF do PERSONAS
FK	CODIGO_BANCO	NUMERIC(12)	Código do banco
	AGENCIA	VARCHAR(10)	Número da agência
	DV_AGENCIA	VARCHAR(2)	Digito verificador da agência
	CONTA	VARCHAR(10)	Numero da conta
	DV_CONTA	VARCHAR(2)	Digito verificador da conta
	TIPO_CONTA	NUMERIC(2)	Poupança ou corrente

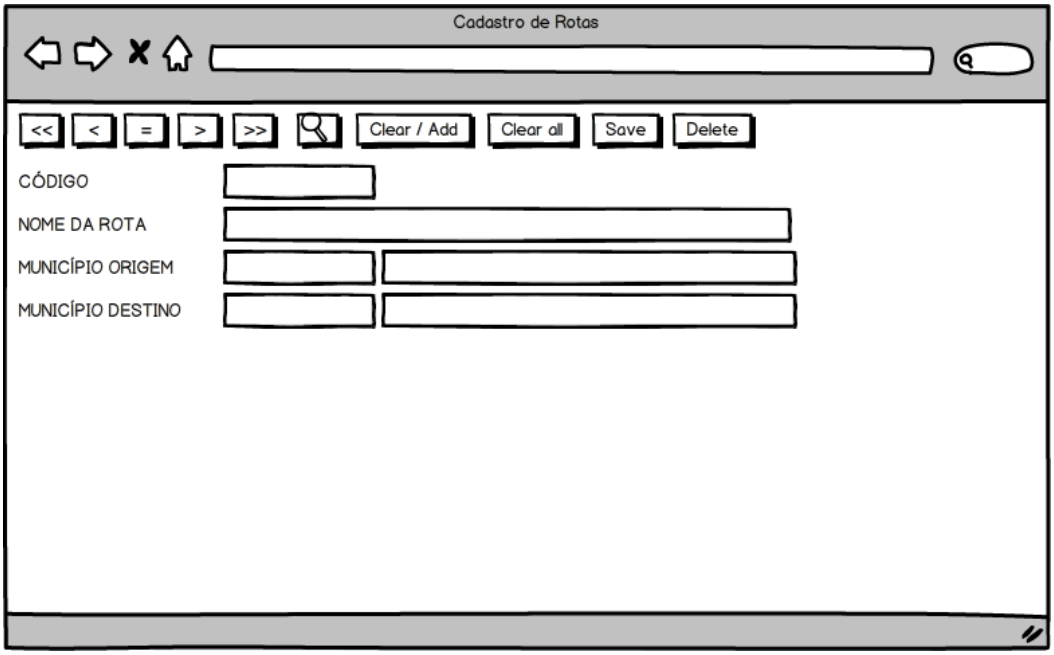
End_Pers – Tabela que contém os dados de endereços dos Personas			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	CNPJ_CPJ	VARCHAR(14)	CNPJ/CPF do PERSONAS
	TIPO_END	CHAR(1)	Tipo de endereço
	CEP	NUMERIC(8)	CEP
	TIPO_LOG	VARCHAR(3)	Tipo de logradouro
	LOGRADOURO	VARCHAR(100)	Logradouro
	NUMERO	VARCHAR(14)	Número
	COMPLEMENTO	VARCHAR(100)	Complemento
	BAIRRO	VARCHAR(100)	Bairro
FK	CODIGO_IBGE_MUNICIPIO	VARCHAR(20)	Código do IBGE do município

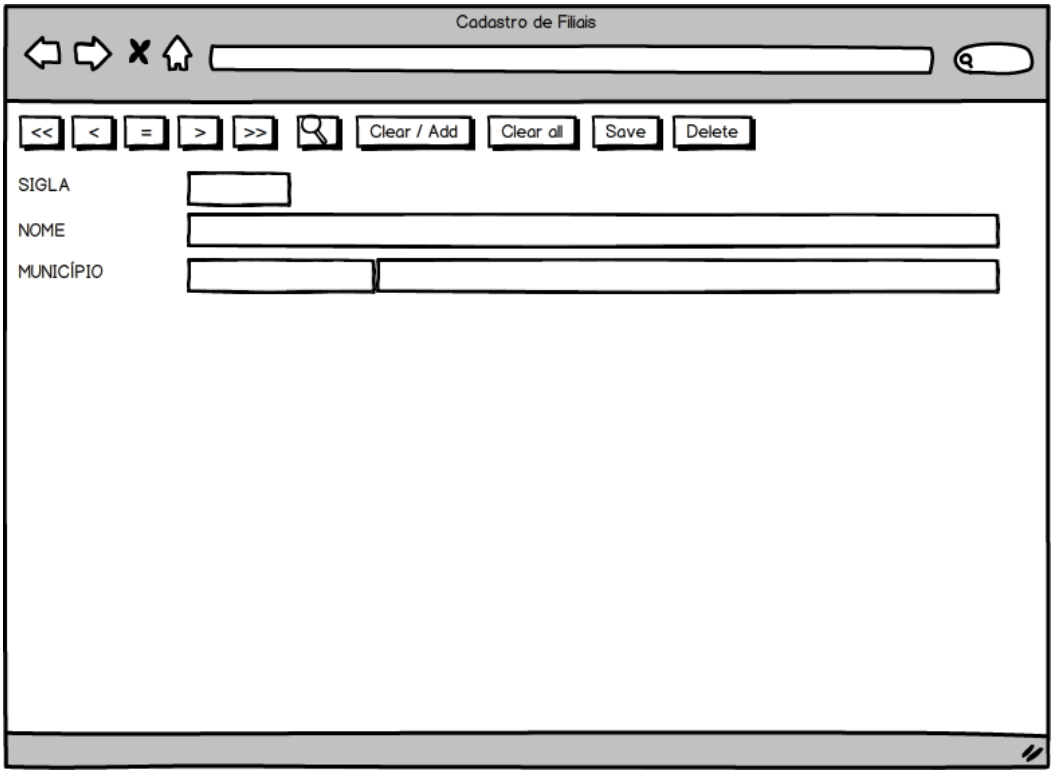
Veiculos – Tabela que contém os dados cadastrais dos veículos			
Chave	**Atributo**	**Tipo**	**Comentário**
PK	PLACA	VARCHAR(7)	Placa do veículo
	TIPO_VEICULO	NUMERIC(2)	Tipo de veículo
	QTDE_EIXOS	NUMERIC()	Quantidade de eixos
	TIPO_RODAGEM	CHARD(1)	Tipo de rodagem
FK	CNPJCPF_PROP	VARCHAR(14)	CNPJ/CPF do proprietário do veiculo
	CATEGORIA	NUMERIC(2)	Categoria do veículo

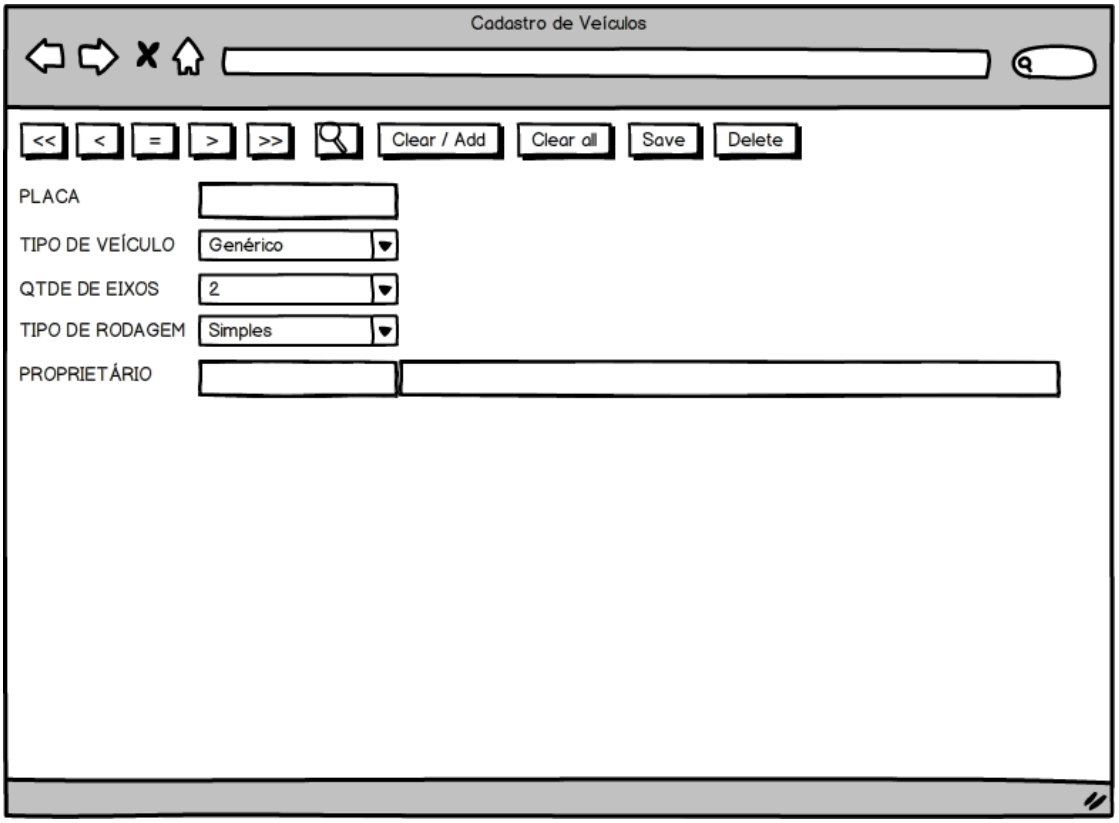
APÊNDICE D – Protótipos dos cadastros






Cadastro de Rotas
<<
<
=
>
>>
Clear / Add
Clear all
Save
Delete
CÓDIGO
NOME DA ROTA
MUNICÍPIO ORIGEM
MUNICÍPIO DESTINO


Cadastro de Filiais
<<
<
=
>
>>
Clear / Add
Clear all
Save
Delete
SIGLA
NOME
MUNICÍPIO


Cadastro de Veículos
<<
<
=
>
>>
Clear / Add
Clear all
Save
Delete
PLACA
TIPO DE VEÍCULO
Genérico
QTDE DE EIXOS
2
TIPO DE RODAGEM
Simples
PROPRIETÁRIO

APÊNDICE E

Abordagem de uma Ferramenta para Comunicação via Web Service com Administradoras de Pagamento Eletrônico de Frete que Atendem a Resolução da ANTT Nº 3.658/11

Gustavo Ferreira Gomes[1], Gustavo Bisognin[2]

[1]Acadêmico do curso de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC - Brasil

[2]Professor do curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brasil.

gustavo@critech.com.br, gbisog@gmail.com

Abstract. *This article refers to software requirements elicitation and logical modeling that seeks to unify the development and integration via Web Service with administratores of electronic payment of freight to attending the brazilian law 3.658/11. Created by the National Agency Land Transportation (ANTT). To realize the research was necessary the study of the disciplines of Software Engineering and Engineering of Requirements such as a technical study on the development of Web Services and an specific knowledge about the actual legislation addressed. After a elicitation of requirements, and also create the logical modeling prototyping, it was possible to develop a standard app that is able to communicate via Web Service with two administratores of electronic payment of freight, this application is simple and effective, for attending the resolution of ANTT.*

Resumo. *O presente artigo refere-se ao levantamento de requisitos de software e modelagem lógica com abordagem de protótipo que visa unificar o desenvolvimento e forma de integração via Web Service, com administradoras de Pagamento Eletrônico de Frete que atendem a resolução 3.658/11 criada pela Agência Nacional de Transportes Terrestres (ANTT). Para a realização da pesquisa foi necessário o estudo das disciplinas de Engenharia de Software e Engenharia de Requisitos, bem como um estudo técnico no desenvolvimento de Web Services e uma busca avançada pelo conhecimento da legislação abordada. Após um levantamento de requisitos, e também de criar a modelagem lógica com prototipação, foi possível desenvolver um aplicativo padrão capaz de comunicar-se via Web Service com duas administradoras de pagamento eletrônico de frete, de forma simples e eficaz, atendendo desta forma a resolução vigente publicada pela ANTT.*

1. Introdução

Decorrente da falta de tempo e entendimento dos requisitos presentes em softwares, muitos analistas acabam projetando alterações e criações de processos desunificados, totalmente isolados, e que se bem elicitados poderiam ser unificados e padronizados, amenizando e evitando custos e investimentos de tempos com mão de obra, visto que quando isso ocorre muitas outras alterações ficam pendentes, e o tempo se torna escasso para execução de rotinas necessárias no sistema.

A necessidade da abordagem do assunto do presente trabalho surgiu após a publicação da resolução nº 3.658/11, criada pela ANTT, em abril de 2011.

Segundo a empresa Dbtrans (2012), com a proibição da carta-frete (sistemática ineficiente de pagamento de frete) e a regulamentação das administradoras de pagamento do transporte de carga, em abril de 2011 através da resolução 3.658/11 da ANTT, os caminhoneiros e transportadoras estão unindo esforços para adaptarem-se as novas obrigações exigidas na resolução.

As regras elaboradas pela Agência Nacional de Transportes Terrestres criaram a figura das administradoras para fazer o vinculo entre o contratante do frete (transportadora), o caminhoneiro (terceirizado contratado) e o órgão regulador (ANTT). Para realizar este vinculo cada Administradora de Pagamento Eletrônico de Frete criou Web Services específicos e disponibilizaram layouts para estes Web Services.

Devido à existência de particularidades em cada administradora de pagamento eletrônico de frete a uma grande preocupação com o tempo gasto e com a qualidade no desenvolvimento e atualizações de softwares comerciais que integram com estas administradoras de PEF. Analisando a definição do problema, observa-se certo grau de dificuldade do desenvolvedor em cumprir os prazos devido às características destas administradoras, e como trata-se de legislação é imprescindível que os prazos estipulados pela ANTT na resolução sejam rigorosamente respeitados.

Motivado e diante do problema exposto, o presente estudo propõe, o levantamento de requisitos de software e modelagem lógica com abordagem de protótipo que unifique o desenvolvimento e forma da integração via Web Service (WS), de duas administradoras de Pagamento Eletrônico de Frete (PEF), respectivamente Apisul e Dbtrans, que atendem a resolução 3.658/11 criada pela Agência Nacional de Transportes Terrestres (ANTT).

2. Engenharia de Software

De acordo com Sommerville (2003, p. 6), “engenharia de software é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema”.

A engenharia de software, além de tornar ampla as garantias relacionadas à qualidade do produto, é a área de conhecimento específico que dá auxílio ao desenho de soluções de software que é utilizado para dar resposta a complexidade crescente e a diversidade dos sistemas, assim como as alterações nos requisitos durante o seu processo de desenvolvimento.

3. Engenharia de Requisitos

Pressman (2006) afirma que compreender os requisitos de um problema é uma das atividades mais difíceis de um engenheiro de software, mas levando em consideração o pensar e agir, o entendimento torna-se mais fácil. Este mesmo autor também afirma que a construção de um projeto de software é desafiadora, criativa e prazerosa, tanto que em muitos casos mesmo que os técnicos não possuam todas as informações necessárias, os mesmos ficam ansiosos para iniciar o desenvolvimento do software.

A engenharia de requisitos, bem como as demais atividades do processo, precisa ser adaptada às necessidades do processo, da tarefa, do projeto, das pessoas envolvidas e, principalmente do produto. Pois estas atividade são de suma importância o desenvolvimento do projeto.

4. Web Services

Segundo Potts (2003, p.3), “um Web Service é uma aplicação de software que pode ser acessada remotamente usando diferentes linguagens baseadas em XML”.

A promessa dos Web Services é baseada na interoperabilidade, ou seja, toda aplicação de software no mundo pode se comunicar com qualquer outra. Ultrapassando todas as antigas barreiras de local, sistema operacional, linguagem, protocolo, entre outros.

Os Web services se baseiam no envio de mensagens eXtensible Markup Language (XML) em um formato Simple Object Access Protocol (SOAP) específico. Mas como a interoperabilidade dos Web Service seja difícil de ser alcançada, as organizações especificadoras buscam a definição de padrões para atingir esta meta.

5. Exigências da ANTT – Resolução nº 3.658/11

A Agência Nacional de Transportes Terrestres, necessitando garantir a movimentação de bens em cumprimento a padrões de eficiência e a modicidade nos fretes no Brasil, e também levando em consideração os problemas causados no mercado de transporte rodoviário de cargas devido às sistemáticas ineficientes de pagamento de frete, decidiu através do artigo 1º da resolução nº 3.658/11 (BRASIL, 2011) regulamentar o pagamento do valor do frete referente à prestação dos serviços de transporte rodoviário de cargas, onde já está previsto no artigo 5º-A da lei nº 11.422, de 2007.

Após regulamentar a resolução que padronizava o pagamento do frete referente à prestação de serviços de transporte rodoviário de cargas, a Agência Nacional de Transportes Terrestres passou a homologar administradoras de meios de pagamento de eletrônico de frete (BRASIL, 2011).

As administradoras de pagamento eletrônico de frete são as empresas habilitadas pela ANTT, responsáveis por realizar o trâmite de forma segura e eletrônica, do pagamento referente a serviço de transporte rodoviário que o contratante faz ao contratado responsável por realizar o frete.

Conforme indica a página web oficial da ANTT, existem atualmente cerca de 20 Administradoras de Meios de Pagamento Eletrônico de Frete habilitadas pelo órgão.

6. Abordagem da Ferramenta de Integração

O trabalho disposto neste presente artigo tem como objetivo realizar a abordagem de uma ferramenta para comunicação via Web Service com Administradoras de Pagamento Eletrônico de Frete, visando atender a resolução nº 3.658/11 disposta pela Agência Nacional de Transportes Terrestres.

A metodologia do trabalho desenvolvido está disposta em 5 etapas, sendo as seguintes: levantamento de requisitos, modelagem lógica, prototipação, implementação e resultados obtidos.

Para realização de cada etapa se fez necessário o estudo das ferramentas a serem utilizadas, onde as mesmas serão relacionadas na próxima seção deste artigo.

6.1 Recursos utilizados nas etapas do desenvolvimento

Na fase de elicitação ou levantamento dos requisitos utiliza-se apenas a versão 2007 do Outlook para comunicar-se com a equipe de suporte a integrações das administradoras e Word para criar a documentação dos requisitos que levantou-se no decorrer desta etapa. Ambos os produtos são da Microsoft Corporation.

A modelagem lógica foi criada utilizando um produto da empresa norte-americana Embarcadero Technologies, o ER/Studio. Para a prototipação das interfaces a ferramenta adotada foi o Balsamiq Mockups, registrado pela empresa italiana Balsamiq Studios. No

processo de implementação o ambiente de desenvolvimento definido foi o Visual DataFlex, versão 17.1, da empresa Data Acess Wordwide.

6.2 Levantamento de requisitos

Nesta primeira etapa do projeto executaram-se alguns passos para elicitação de todos os requisitos do processo de comunicação via Web Service com as administradoras.

No primeiro passo foram solicitados os manuais atualizados das três administradoras mencionadas respectivamente no capítulo cinco (Dbtrans, Pamcary e Apisul), onde se definiu que o levantamento seria realizado com base nos manuais disponibilizados pelas administradoras Dbtrans e Apisul, devido às mesmas possuírem uma documentação mais acessível dos Web Services oferecidos.

Após definição das administradoras, realizou-se um estudo de quais serviços são necessários consumir em cada Web Service para atender a resolução disposta pela ANTT, visto que as administradoras disponibilizam serviços adicionais que não são necessários para atender a resolução. Ficou concluído que no mínimo devem ser utilizados os seguintes serviços para atendimento da resolução, conforme apresentados na tabela 1.

Tabela 1. Métodos básicos para atendimento da resolução

Administradora	Métodos
Dbtrans	Autenticar cliente; Manter Viagem; Manter Transportador; Manter Veículo; Consultar Rotas.
Apisul	Declaração de Operação de Transporte.

A partir dos métodos básicos necessários vistos na tabela 1 realizou-se um detalhamento de cada um dos serviços. Na seqüência foi solicitado as equipes de suporte de cada administradora de PEF, os dados de autenticação em ambiente de homologação para posterior comunicação via Web Service.

Para finalização da elicitação de requisitos definiu-se o ambiente que o aplicativo iria ser executado em ambiente web.

6.3 Modelagem lógica do banco de dados

A criação da modelagem lógica do sistema é essencial para que seja criada uma ferramenta de comunicação padrão, com banco de dados e aplicação, ambos sem redundâncias, desta forma é possível atender a integração com no mínimo duas administradoras PEF utilizando o mesmo banco de dados.

As tabelas, campos, definição de tipos de campos, verificação de primary key e foreign key foram modeladas com base no elicitação descrita na seção anterior deste artigo e visando a próxima etapa de prototipação e desenvolvimento.

Representado pelo modelo Entidade-Relacionamento a seguir (figura 1), o modelo lógico foi implementado no software ER/Studio.

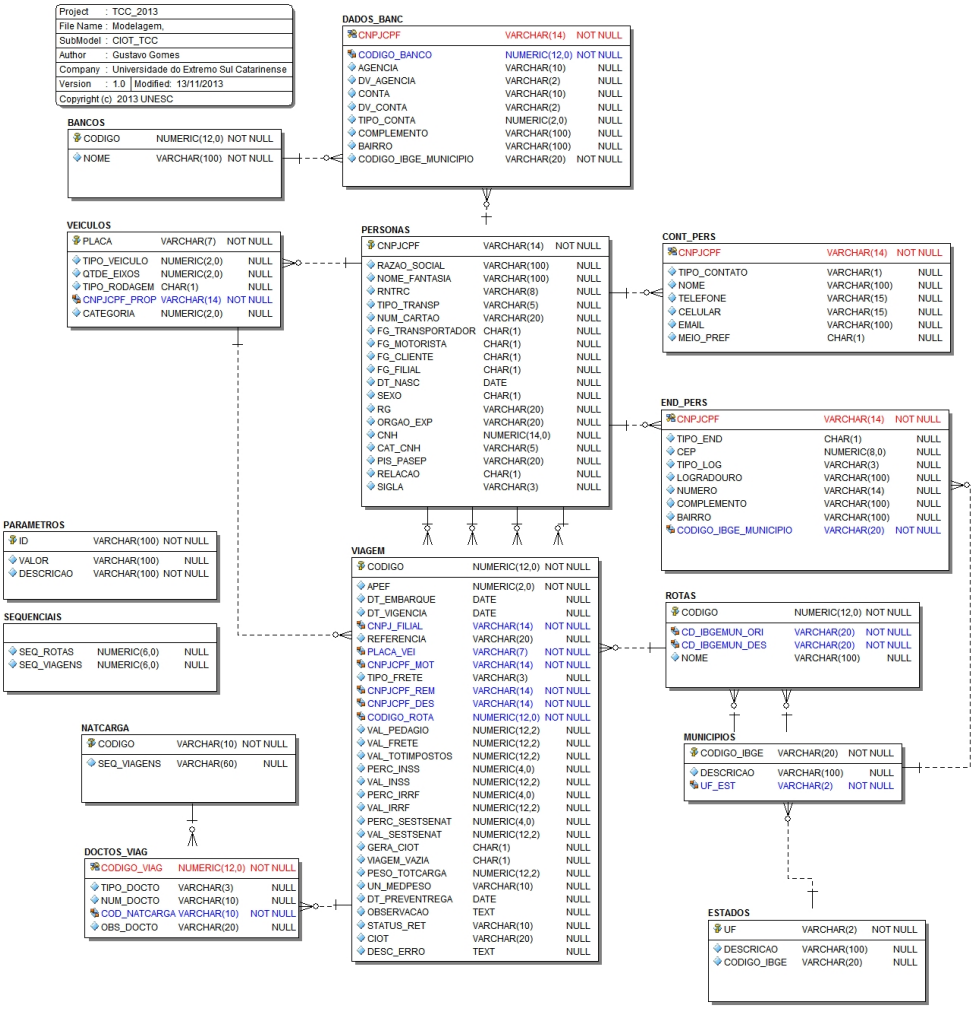


Figura 1. Modelo Entidade-Relacionamento

Os campos criados na modelagem da figura anterior são idôneos para no mínimo atender a resolução 3.658/11 utilizando a administradora Dbtrans ou Apisul, independente da administradora de PEF que está sendo utilizada, porém para a percepção concreta deste padrão foram criadas as etapas de prototipação e por seqüente a implementação.

6.4 Prototipação das telas

Assim como realizou-se na seção de modelagem lógica, a presente seção apresentará na figura 2 o protótipo da principal janela da ferramenta, utilizando o aplicativo Balsamiq Mockups.

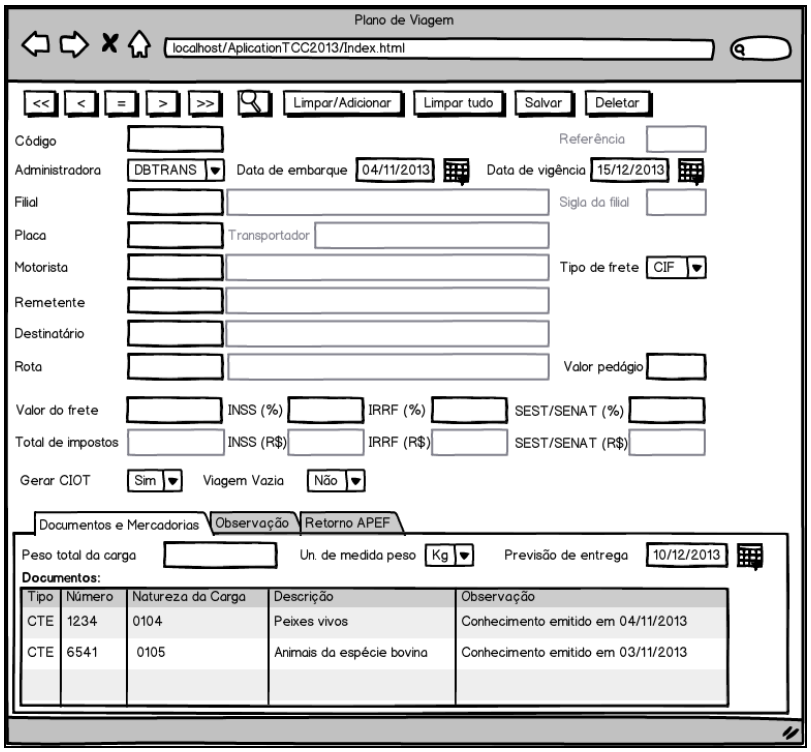


Figura 2. Protótipo gráfico da tela de plano de viagem

Após apresentação do design gráfico da tela mostrada na figura 2 e análise dos campos da tela conforme levantamento dos requisitos, serão listados na tabela 2 uma breve descrição de cada componente da tela, com algumas regras de interface.

Tabela 2. Regras de interface

Componente	Regra/descrição
Código	Auto-incremento. Código utilizado para consultar um plano de viagem.
Referência	Inacessível. Mostrar neste campo a sigla da filial concatenada ao código da viagem.
Administradora	DBTRANS é a opção sugerida. As opções existentes na combo são DBTRANS e APISUL.
Data de embarque	A data deve ser maior ou igual à data atual.
Data de vigência	A data deve ser maior ou igual à data atual e de embarque.
Filial	Após informar o CNPJ da filial os campos de nome e sigla da filial devem ser mantidos inacessíveis.
Placa	Após informar a placa o campo de transportador deve ser mantido inacessível.
Motorista	Após informar um CPF de motorista o campo de nome deve ser mantido inacessível.
Tipo de frete	CIF é a opção sugerida. As opções existentes na combo são CIF (pagador do frete é o remetente) e FOB (pagador do frete é o destinatário).

Remetente	Após informar o CNPJ do cliente remetente o campo de nome deve ser mantido inacessível.
Destinatário	Após informar o CNPJ do cliente destinatário o campo de nome deve ser mantido inacessível.
Rota	Após informar o código da rota o campo de nome de ver mantido inacessível.
Valor do frete	O valor do frete deve ser maior que zero.
INSS (%)	O percentual de INSS retido deve ser maior que zero.
IRRF (%)	O percentual de IRRF retido deve ser maior que zero.
SEST/SENAT (%)	O percentual de SEST/SENAT retido deve ser maior que zero.
Total de impostos	Este campo sempre estará inacessível e mostrará automaticamente o valor total de impostos conforme valores mostrados nos campos de INSS (R$), IRRF (R$) e SEST/SENAT (R$).
INSS (R$)	Este campo sempre estará inacessível e mostrará o valor do INSS retido de acordo com o valor do frete e do percentual de imposto informado.
IRRF (R$)	Este campo sempre estará inacessível e mostrará o valor do IRRF retido de acordo com o valor do frete e do percentual de imposto informado.
SEST/SENAT (R$)	Este campo sempre estará inacessível e mostrará o valor do SEST/SENAT retido de acordo com o valor do frete e do percentual de imposto informado.
Gerar CIOT	Sim é a opção sugerida. As opções existentes na combo são Sim e Não.
Viagem Vazia	Não é a opção sugerida. As opções existentes na combo são Sim e Não.
Documentos e Mercadorias	Nesta aba devem ser preenchidos os documentos que comprovam o transporte da carga, bem como o peso, a unidade de medida do peso e a previsão de entrega das mercadorias. A previsão de entrega deve ser maior ou igual a data atual.
Observação	Na aba de Observação poderá ser informada uma observação qualquer para viagem.
Retorno APEF	Na aba APEF serão mostrados os dados de retorno logo após que for salvo o plano de viagem e a comunicação ser realizada com a administradora. Os dados devem ser mostrados de forma inacessível, sendo que poderão ser visualizados nesta aba os campos de status, CIOT e descrição do erro retornado.

As regras de interface descritas na tabela 2 podem evitar a falha de comunicação no momento que a administradora de pagamento eletrônico de frete retornar uma requisição, ou seja, quando há falha de comunicação as administradoras simplesmente retornam a mensagem vazia. As demais regras para geração do CIOT e conseqüente cumprimento da resolução 3.658/11 ficam a cargo de cada administradora.

Com a prototipação, regras de interface, e demais etapas citadas nas seções anteriores da metodologia concluídas, a próxima seção abordará a implementação da ferramenta.

6.5 Implementação do aplicativo

A implementação é a última etapa da ferramenta abordada no presente artigo, onde se aplica todas as outras principais etapas citadas nas seções anteriores na qual pertencem a áreas da engenharia de software. Esta fase de desenvolvimento iniciou-se com a criação do banco, utilizando o próprio banco nativo e disponibilizado pelo Visual DataFlex representado pelo exemplo figura 3.

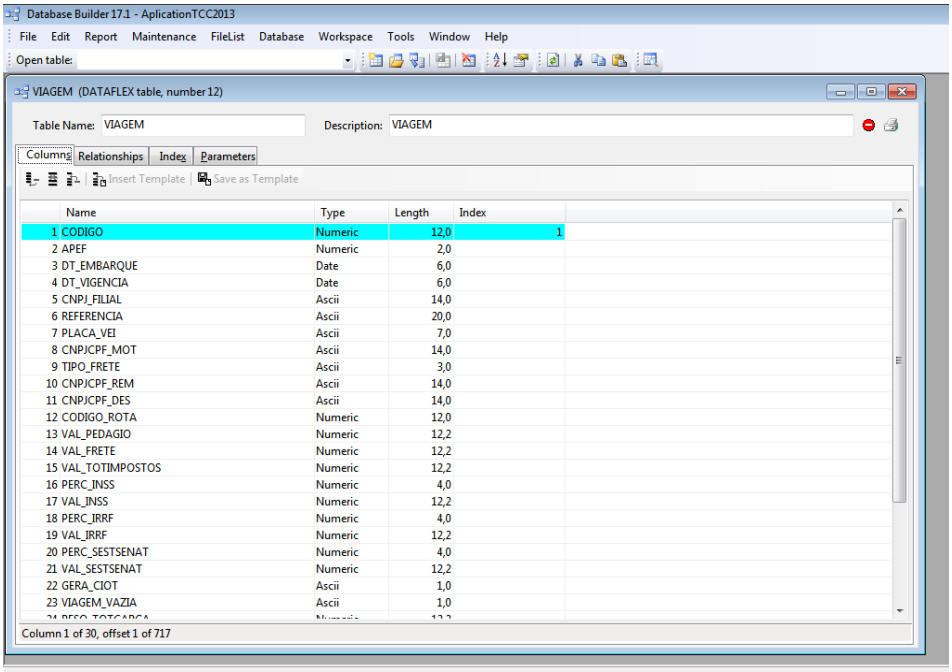


Figura 3. Exemplo de criação de tabela utizando o Visual DataFlex 17.1

Posteriormente foram criados respectivamente o dicionário de dados de cada tabela utilizada e foram desenvolvidas as telas de consulta denominadas na IDE como lookups (pesquisas).

Após a criação da base para desenvolvimento mencionada acima, iniciou-se o processo de integração via Web Service com as administradoras (conforme figura 4) utilizando a mesma estrutura de banco de dados e compartilhando objetos em comum em uma única função.

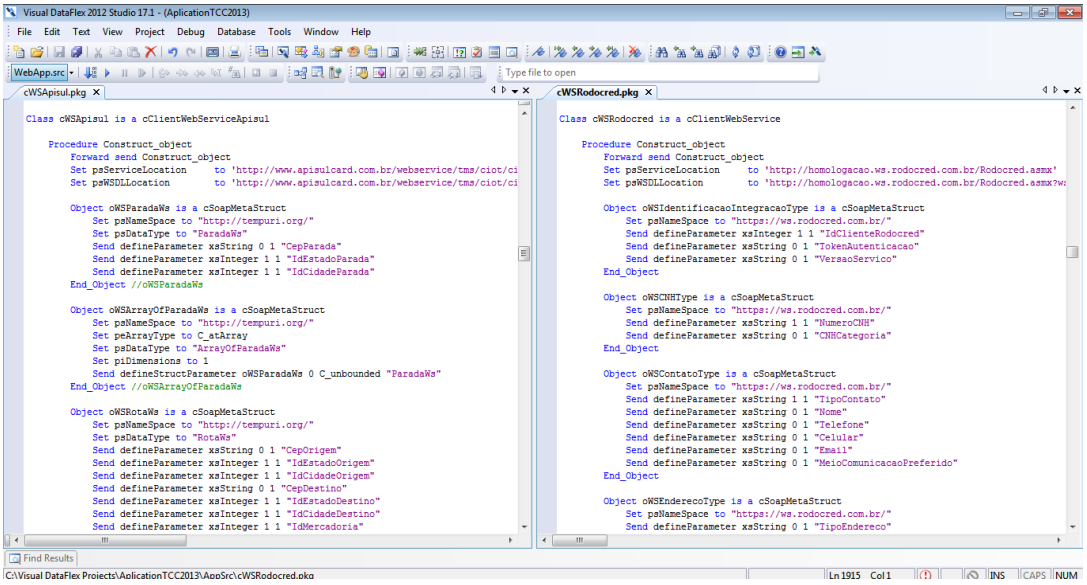


Figura 4. Código fonte das classes das integrações com as administradoras

O desenvolvimento dos pacotes de comunicação foram implementados consumindo o Web Service de homologação de ambas as administradoras a partir do WSDL disponibilizado em cada manual, da Apisul e Dbtrans.

Para permitir que o usuário final tenha acesso ao padrão de integração foram desenvolvidas as telas de cadastros e também a principal tela já mencionada na prototipação, a tela de plano de

viagem. Para chamada de cada interface foi criado um menu disposto na barra superior de modo horizontal.

Esta tela principal denominou-se plano de viagem e pode ser visualizada na figura 5.

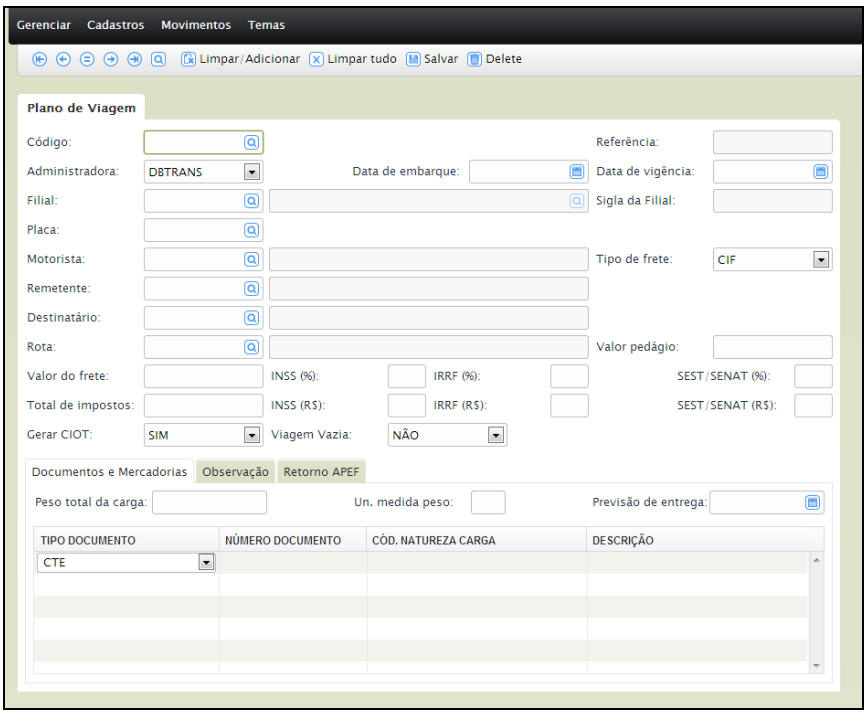


Figura 5. Plano de viagem

Ao utilizar a tela mencionada na figura 5, é possível criar uma viagem apontando qual administradora será realizada para que a resolução 3.658/11 da ANTT seja atendida. Por fim, a resolução vigente é atendida quando o Web Service da administradora de PEF escolhida para o plano de viagem corrente retorna o status de sucesso e um número de CIOT, onde ambos os campos estão dispostos na aba de "Retorno APEF" desta tela.

7. Conclusão

Através do levantamento bibliográfico e elicitação dos requisitos, foi possível definir o que é necessário para criação de um aplicativo que atendesse a resolução 3.658/11 da ANTT independente das particularidades de duas administradoras de pagamento eletrônico de frete.

Utilizando técnicas e modelos da engenharia de software foi possível criar a ferramenta de integração com telas organizadas e limpas, com poucas tabelas se comparado a quantidade de serviços que as administradoras disponibilizam, sendo que os serviços que não foram utilizados, não são relevantes perante a legislação apontada neste trabalho. Com isso, comprovou-se que é possível reduzir o tempo com estes tipos de integrações, que são gastos por equipes de TI das transportadoras ou terceirizadas.

Na realização de testes de comunicação com ambas as administradoras utilizando a tela de plano de viagem, possibilitou-se consumir o número do CIOT, que o código que representa que a resolução 3.658/11 foi atendida com sucesso.

Referências

BRASIL. Resolução nº 3.658, de 19 de Abril de 2011. Diário Oficial [da] República Federativa do Brasil, Poder Executivo, Brasília, DF, 27 abr. 2011. Seção 1, p. 97.

DBTRANS. DBTRANS é habilitada pela ANTT para pagamento eletrônico de frete. Disponível em: <http://www.rodocred.com.br/Imprensa/Releases/DBTRANS%C3%A9habilitadaparapagamentoeletr%C3%B4nico/tabid/518/Default.aspx>. Acesso em: 10 abr. 2013.

DBTRANS. Detalhamento técnico da integração Rodocred: Versão 2.2. Rio de Janeiro: Dbtrans, 2011. 118 p.

POTTS, Stephen; KOPACK, Mike. Aprenda Web services em 24 horas,

tradução de Marcos Vieira. Rio de Janeiro: Campus, 2003. 367 p.

PRESSMAN, Roger S. Engenharia de Software. 6. ed. São Paulo: McGraw-Hill, 2006.

SOMMERVILLE, Ian. Engenharia de Software. São Paulo: Addison-Wesley, 2003. 592 p.