



UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
Câmpus de Presidente Prudente

Fernanda Hondo Tedesque

*Recebimento de dados GNSS via Internet em
RTCM 3 e conversão para RINEX utilizando a
plataforma J2ME*

2011

Fernanda Hondo Tedesque

*Recebimento de dados GNSS via Internet
em RTCM 3 e conversão para RINEX
utilizando a plataforma J2ME*

Trabalho de conclusão de curso apresentada
ao Departamento de Matemática, Estatística
e Computação (DMEC), da Universidade Es-
tadual Paulista “Júlio de Mesquita Filho”,
sob o título **Recebimento de dados GNSS
via Internet em RTCM 3 e conversão
para RINEX utilizando a plataforma
J2ME.**

Orientador: Prof. Dr. Milton Hirokazu Shimabukuro

Presidente Prudente

2011

Resumo

Grandes mudanças estão ocorrendo em todo o mundo, inclusive na área de posicionamento utilizando Sistemas Globais de Navegação por Satélite em aparelhos móveis como: celulares ou laptops. Isso se deve ao grande avanço e disponibilidade dos serviços de internet para esses dispositivos móveis. A precisão e rapidez das transmissões de dados para esses aparelhos torna as tecnologias de envio/recebimento de dados via internet um alvo de estudos para a otimização. A otimização pode permitir a obtenção, em tempo real, de coordenadas (latitude, longitude e altitude) de melhor qualidade por usuários de equipamentos de GPS (*Global Positioning System*) que possuam acesso à internet sem fio, como os utilizados para controle de frota, locomoção, navegação marítima, agricultura, etc. A utilização do protocolo NTRIP (*Networked Transport of RTCM via Internet Protocol*) em aplicações GPS vem crescendo cada dia mais. Aplicações estão disponíveis em C, o que cria uma oportunidade para o desenvolvimento com foco em ambientes multiplataforma. Nesse contexto, é proposto um aplicativo, implementado em um ambiente multiplataforma e com a utilização do NTRIP, capaz de ser executado em um dispositivo móvel para o recebimento de dados GNSS.

Palavras-chave: Sistemas Globais de Navegação por Satélite. NTRIP. RTCM.

Abstract

Changes are occurring worldwide, including the area of positioning using Global Navigation Satellite Systems in mobile devices such as mobile phones or laptops. This is due to the great improvement and availability of Internet services to these mobile devices. The accuracy and speed of data transmission for these devices makes the technology of sending / receiving data via an internet targeted for optimization. The optimization could allow obtaining, in real-time, coordinates (latitude, longitude and altitude) of suitable quality for users of GPS (Global Positioning System) devices that have wireless Internet access, such as those used to control the fleet, locomotion, navigation, agriculture, etc. . The use of the protocol NTRIP (Networked Transport of RTCM via Internet Protocol) in GPS applications is growing every day. Applications are available in C, which creates an opportunity for development with a focus on multiplatform environments. In this context, we propose an application, implemented in a multiplatform environment and the use of NTRIP, able to run on a mobile device for receiving GNSS data.

Keywords: Global Navigation Satellite System. NTRIP. RTCM.

Sumário

Lista de abreviaturas e siglas	p. 11
1 Introdução	p. 13
1.1 Objetivos do Trabalho	p. 14
1.2 Organização do Trabalho	p. 14
2 Fundamentação Teórica	p. 15
2.1 Protocolos da Rede	p. 15
2.1.1 Protocolo	p. 15
2.1.2 Protocolo HTTP	p. 16
2.1.2.1 Conexão	p. 16
2.1.2.2 Servidores e Clientes HTTP	p. 18
2.1.2.3 Mensagens	p. 18
2.1.2.4 Requisições HTTP	p. 19
2.1.2.5 Resposta HTTP	p. 20
2.1.3 Protocolos da camada de Transporte	p. 23
2.1.3.1 Protocolo TCP	p. 23
2.1.3.2 Mecanismos TCP	p. 23
2.1.3.3 Estrutura de segmento TCP	p. 24

2.1.3.4	Serviço Orientado à Conexão	p. 26
2.1.3.5	Protocolo UDP	p. 26
2.1.3.6	Estrutura de segmento do UDP	p. 27
2.2	NTRIP	p. 28
2.2.1	Protocolo NTRIP	p. 28
2.2.2	Componentes do NTRIP	p. 29
2.2.3	Ntrip Server	p. 29
2.2.3.1	Janelas de ajustes	p. 30
2.2.3.2	Comunicação NtripServer para NtripCaster	p. 31
2.2.4	Ntrip Caster	p. 32
2.2.5	Ntrip Client	p. 32
2.2.5.1	Comunicação NtripClient para NtripCaster no NTRIP	p. 33
2.3	RTCM	p. 36
2.4	Padrão RTCM	p. 36
2.4.1	RTCM 2.3	p. 36
2.4.2	RTCM 3.1	p. 38
2.4.3	Estrutura RTCM 3	p. 41
2.5	Formato RINEX	p. 42
2.5.1	Especificação das Observáveis	p. 42
2.5.2	Versões	p. 43
2.5.3	Especificações dos arquivos RINEX	p. 44
2.6	J2ME - JAVA Micro Edition	p. 45
2.6.1	Dispositivos Móveis	p. 45
2.6.2	Plataforma JAVA	p. 46
2.6.3	Arquitetura J2ME	p. 47

3.1	Considerações iniciais	p. 50
3.2	Desenvolvimento	p. 50
3.2.1	Ferramentas utilizadas	p. 50
3.2.2	Arquitetura da Aplicação	p. 52
3.2.2.1	MIDlets	p. 52
3.2.2.2	Arquitetura	p. 52
3.3	Testes	p. 57
4	Conclusões	p. 61
4.1	Trabalhos Futuros	p. 62
	Referências	p. 63
	Apêndice A – Arquivos RINEX gerados pela aplicação	p. A.1
	Apêndice B – Tipos de mensagens RTCM 3 abordadas nesse projeto	p. B.1

Lista de Figuras

1	Modelo de camadas do TCP/IP (COMER, 2007).	p. 16
2	Sockets TCP e processos que os utilizam para comunicação (KUROSE J. F. E ROSS, 2005).	p. 17
3	Mensagens de requisição e respostas HTTP (KUROSE J. F. E ROSS, 2005).	p. 18
4	Formato geral de uma mensagem de requisição (KUROSE J. F. E ROSS, 2005).	p. 19
5	Típica mensagem de requisição HTTP (KUROSE J. F. E ROSS, 2005).	p. 20
6	Formato geral de uma mensagem de resposta (KUROSE J. F. E ROSS, 2005).	p. 21
7	Típica mensagem de resposta HTTP (KUROSE J. F. E ROSS, 2005).	p. 22
8	Estrutura de segmento TCP (KUROSE J. F. E ROSS, 2005).	p. 24
9	Estrutura de segmento do UDP (KUROSE J. F. E ROSS, 2005).	p. 27
10	Estrutura e relacionamento dos componentes do NTRIP (RTCM, 2009).	p. 30
11	Janelas de ajustes no NtripServer (DETTMERING; WAESE; WEBER, 2006).	p. 30
12	Janela de ajuste para conexão automática (DETTMERING; WAESE; WEBER, 2006).	p. 31
13	Mensagem de requisição NtripServer para NtripCaster (RTCM, 2009).	p. 31
14	Mensagem de resposta NtripCaster para NtripServer (RTCM, 2009).	p. 32
15	Exemplo de Sourcetable (RTCM, 2005).	p. 33
16	Dispositivo, que roda o componente NtripClient, mostrando em tela alguns <i>mountPoints</i> de uma <i>Sourcetable</i> (GMBH, 2010).	p. 34

17	Mensagem de requisição da Tabela(<i>Sourcetable</i>) para NtripCaster (RTCM, 2009).	p. 34
18	Mensagem de resposta com a Tabela(<i>Sourcetable</i>) para NtripClient (RTCM, 2009).	p. 34
19	Mensagem de requisição NtripClient para NtripCaster (RTCM, 2009). .	p. 35
20	Mensagem de resposta NtripCaster para NtripClient (RTCM, 2009). . .	p. 35
21	Tipos de mensagens do RTCM 2.3 (DALBELO, 2010).	p. 37
22	Cabeçalho da mensagem RTCM 2.3 com suas duas primeiras palavras. Fonte:(DALBELO, 2010).	p. 37
23	Indicador de saúde da estação (DALBELO, 2010).	p. 38
24	Grupo de Mensagens do RTCM 3. Traduzido de (RTCM, 2006).	p. 39
25	Cabeçalho das mensagens RTCM 1001-1004. Traduzido de (RTCM, 2006). .	p. 40
26	Estrutura de uma mensagem do formato RTCM versão 3. Traduzido de (RTCM, 2006).	p. 41
27	Cabeçalho de um arquivo de observação RINEX 3.0 (GURTNER; ESTEY, 2007).	p. 44
28	Plataforma Java (ROMEIRO, 2005).	p. 46
29	Camadas da arquitetura J2ME (MUCHOW, 2007).	p. 47
30	Ambientes da plataforma Java e os Perfis da J2ME. Adaptado de (JOHNSON, 2007).	p. 49
31	Netbeans IDE 6.8.	p. 51
32	Ciclo de vida de um MIDlet (JOHNSON, 2007).	p. 52
33	Diagrama das principais partes da aplicação.	p. 53
34	Conexão utilizando socket.	p. 53
35	Trecho de código do envio da mensagem de requisição e recebimento dos dados.	p. 54
36	Trecho de código que realiza a conversão dos dados.	p. 55
37	Parte inicial da definição da classe RTCM3ParserData	p. 55

38	Mensagem do tipo 1004 do RTCM 3.	p. 56
39	Trecho de código que realiza a formatação de strings e grava no arquivo.	p. 57
40	Emulador.	p. 57
41	Resultado: recebimento da Sourcetable.	p. 58
42	Procedimentos para a geração do Arquivo RINEX.	p. 59
A.1	Arquivo de dados de Observação RINEX 2.11	p. A.2
A.2	Arquivo de dados de Observação RINEX 3.0	p. A.3
A.3	Arquivo de Mensagem NAVEGAÇÃO GPS RINEX 2.10	p. A.4
A.4	Arquivo de Mensagem NAVEGAÇÃO GLONASS RINEX 2.10	p. A.5
A.5	Arquivo de Mensagem de Navegação GNSS Rinex 3.0 (Mixed GPS / GLONASS). Este arquivo contém dados sobre dois sistemas de satélites: o GPS e o GLONASS, comprovando que os arquivos RINEX 3 podem conter dados de mais de um sistema de satélite.	p. A.6
B.1	Cabeçalho das mensagens 1001-1004. (RTCM, 2006)	p. B.1
B.2	Conteúdo da mensagem 1001. (RTCM, 2006)	p. B.2
B.3	Conteúdo da mensagem 1002. (RTCM, 2006)	p. B.2
B.4	Conteúdo da mensagem 1003. (RTCM, 2006)	p. B.3
B.5	Conteúdo da mensagem 1004. (RTCM, 2006)	p. B.3
B.6	Cabeçalho das mensagens RTCM 1009-1012. (RTCM, 2006)	p. B.4
B.7	Conteúdo da mensagem 1009. (RTCM, 2006)	p. B.4
B.8	Conteúdo da mensagem 1010. (RTCM, 2006)	p. B.5
B.9	Conteúdo da mensagem 1011. (RTCM, 2006)	p. B.5
B.10	Conteúdo da mensagem 1012. (RTCM, 2006)	p. B.6
B.11	Conteúdo da mensagem 1019. (RTCM, 2006)	p. B.7
B.12	Conteúdo da mensagem 1020. (RTCM, 2006)	p. B.8

Lista de Tabelas

1	Comparação entre as precisões das coordenadas resultantes	p. 60
---	---	-------

Lista de abreviaturas e siglas

AWT Abstract Windowing Toolkit

BKG Bundensamt für Katrographie und Geodäsie

CDC Connected Device Configuration

CLDC Connected, Limited Device Configuration

GNSS Global Navigation Satellite Systems

GPS Global Positioning System

HTTP Hypertext Transfer Protocol

IP Internet Protocol

J2ME Java 2 Micro Edition

JEE Java Enterprise Edition

JSE Java Stantard Edition

JVM Java Virtual Machine

KVM Kilo Virtual Machine

MIDP Mobile Information Device Profile

NTRIP Networked Transport of RTCM via Internet Protocol

PDA Personal Digital Assistant

RINEX Receiver Independent Exchange Format

RTCM Radio Technical Commission for Maritime Services

RTK Real Time Kinematic

TCP Transmission Control Protocol

UDP User Datagram Protocol

Introdução

Nos últimos anos, o avanço da internet tornou-se um grande atrativo para pesquisas e estudos. O crescimento contínuo da Internet global é um dos fenômenos mais interessantes e excitantes em redes. Há pouco mais de vinte anos, a Internet era um projeto de pesquisa que envolvia algumas dúzias de sites (COMER, 2007). O mundo todo é capaz de comunicar-se e interagir através da rede Internet, e muitas outras vantagens fazem com que novas ideias surjam para explorar todos benefícios que essa rede oferece.

Da mesma maneira, muitos experimentos e muitas pesquisas tem sido desenvolvidas para realização da transmissão de dados GNSS (*Global Navigation Satellite Systems*) pela Internet. Essas pesquisas possibilitaram o surgimento do protocolo chamado NTRIP (*Networked Transport of RTCM via Internet Protocol*) que efetua a disseminação dos dados pela rede. Esses dados podem ser coletados por qualquer dispositivo, móvel ou não, que seja capaz de realizar uma conexão. Os dados, após serem recebidos, podem ser empregados de diversas maneiras, de acordo com o que o usuário necessite, sendo possível a conversão desses dados RTCM para arquivos texto RINEX, por exemplo. Detalhes sobre RTCM e RINEX são encontrados no capítulo 2.

Simultaneamente a esses acontecimentos, os dispositivos móveis, que utilizam da tecnologia de comunicação sem fio, têm gerado grande motivação para desenvolvimento de novas aplicações, decorrente ao crescente mercado desses aparelhos. Diversos serviços podem ser implementados para computação móvel. Em geral, temos acesso à troca de mensagens (SMS, e MMS), chats, correio eletrônico(...) (JOHNSON, 2007).

Sendo assim, é válido considerar a possibilidade de se obter um aplicativo que efetue a conversão de dados, recebidos via internet, em arquivos RINEX e que rode em um

dispositivo móvel como um celular.

1.1 Objetivos do Trabalho

A intenção desse projeto consiste em obter bases para implementação e testes de um aplicativo, em ambiente multiplataforma, apartir de aplicativos já existentes desenvolvidos na plataforma C mas que não podem ser executados em dispositivos móveis. Esse projeto tem como objetivo principal o desenvolvimento e a implementação de um aplicativo, para dispositivos móveis, que realize conexão com a internet e, utilizando o protocolo NTRIP, seja capaz de receber mensagens no formato RTCM e, posteriormente, converter e escrever essas mensagens em arquivo RINEX.

1.2 Organização do Trabalho

O restante do documento é organizado como segue:

No Capítulo 2 discorre-se sobre os protocolos da rede Internet, evidenciando o importante papel e as funções que cada protocolo desempenha na rede. É feita a explanação do protocolo NTRIP, que é o protocolo utilizado para o envio das mensagens/dados RTCM através da internet. O padrão RTCM também é discutido, abordando-se suas versões e as estruturas que suas diferentes mensagens podem conter e, em seguida, é apresentado o formato RINEX e os detalhes de suas versões. Por fim, discorre-se sobre a plataforma J2ME, que é o ambiente Java utilizado para o desenvolvimento deste projeto.

No Capítulo 3 apresenta-se o processo adotado no desenvolvimento do aplicativo. São justificadas as escolhas entre as soluções encontradas para a implementação da ferramenta, e, então, é apresentada a arquitetura do aplicativo e seu funcionamento.

No Capítulo 4 é apresentada a conclusão obtida após o desenvolvimento deste projeto e é feita uma observação para os futuros trabalhos.

Capítulo 2

Fundamentação Teórica

2.1 Protocolos da Rede

Nesta Seção é dada ênfase aos protocolos, presentes nas camadas da rede, que prestam serviços em conjunto, e sistematizam a transmissão de pacotes sobre a Internet. Esses protocolos atuam juntos, complementando um ao outro, para que suas funções sejam efetuadas com sucesso. Assim, as características e o papel de cada protocolo serão apresentados mais detalhadamente mostrando suas interações com o NTRIP e sua importância para a realização da transmissão.

2.1.1 Protocolo

Um protocolo da rede pode ser definido como um conjunto de regras que regem a comunicação entre duas ou mais entidades que desejam trocar informações. Um protocolo define o formato e a ordem das mensagens trocadas entre duas ou mais entidades comunicantes, bem como nas ações realizadas na transmissão e/ou no recebimento de uma mensagem ou outro evento (KUROSE J. F. E ROSS, 2005).

Existem variados tipos de protocolos que atuam em diferentes camadas existentes em uma rede. A organização em camadas foi estabelecida a fim de ocultar a complexidade das operações em que os dados passam quando percorrem todo o caminho até chegar ao destino. A divisão em camadas é uma idéia poderosa porque simplifica o projeto e o teste de protocolos (COMER, 2007). Existe uma hierarquia entre essas camadas, em que cada camada presta serviços à camada imediatamente acima dela. Para isso, existem protocolos residentes em cada camada que implementam funcionalidades relativas à ela,

e tornam os serviços desta camada mais sistematizados.

O NTRIP fornece dados através de redes móveis de arquitetura TCP/IP - advém dos dois famosos protocolos da rede Internet - o TCP (*Transmission Control Protocol*) que é responsável pelo transporte dos dados, e o IP (*Internet Protocol*) que é o responsável por encaminhar os dados pela rede. Esse modelo contém cinco camadas como mostra a Figura 1. O NTRIP utiliza desse modelo de camadas para obter os benefícios por ela oferecidos, contando com o serviço de alguns protocolos importantes atuantes na rede Internet.

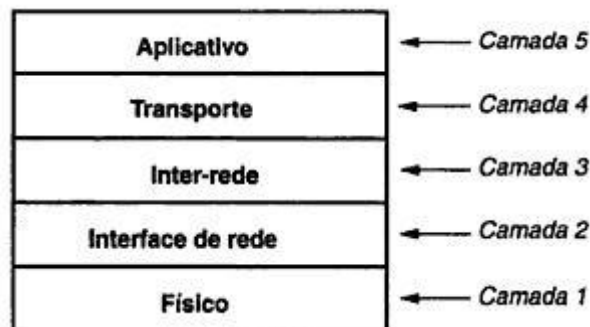


Figura 1: Modelo de camadas do TCP/IP (COMER, 2007).

2.1.2 Protocolo HTTP

O protocolo HTTP - Protocolo de Transferência de Hipertexto (*Hypertext Transfer Protocol*) - reside na camada de aplicação da pilha de protocolos TCP/IP e pode ser considerado a base de sustentação da web. Ele é usado para distribuição e recuperação de informações na *World Wide Web*. O NTRIP se baseia no protocolo HTTP/1.1 definido no RFC-2616 que é a versão mais atual, embora muitos browsers e servidores ainda utilizem a versão 1.0, sendo essas versões compatíveis entre si. O HTTP pode ser dividido entre programa cliente e programa servidor, atuando como um conjunto de regras que conduzem o diálogo entre essas entidades comunicantes. Basicamente, esse protocolo consiste em requisições ou mensagens de pedido e respostas no formato ASCII feitos em comandos com vocábulos da língua inglesa.

2.1.2.1 Conexão

O protocolo de transporte utilizado pelo HTTP é o TCP, e através de *sockets* existentes nos processos cliente e servidor, o HTTP efetua, com a conexão TCP, a troca de informações de forma confiável, sendo esse o motivo do UDP não ser o protocolo escolhido para realizar esse serviço.

Socket é a interface do processo com a rede: nos processos cliente e servidor são como portas por onde as mensagens de requisições e respostas fluem, veja Figura 2. O *socket* é basicamente um recurso na programação cliente-servidor que atua como rotinas que permitem a interligação e transferência de informações entre as entidades comunicantes. Quando cria um *socket*, um aplicativo deve especificar a família de protocolos a ser usada, como também o tipo de serviço desejado (orientado à conexão ou sem conexão). Depois de criar um *socket*, o servidor especifica um endereço local para o *socket* e o tamanho da fila de requisição. Depois de criar um *socket*, o cliente especifica o endereço de um servidor remoto (COMER, 2007).

Por padrão, as conexões TCP utilizadas pelo HTTP são persistentes, ou seja, elas permanecem abertas durante a transferência de todos os dados e são fechadas quando troca de informações é suficiente e nada mais é requerido. Enquanto a conexão permanece aberta, várias requisições podem ser feitas, sequencialmente ou paralelamente, e vários objetos podem ser solicitados e respondidos nessa mesma conexão. Porém, as conexões podem ser configuradas de acordo com o que é conveniente para a transmissão. No NTRIP, a parte HTTP ainda não é capaz de suportar conexão persistente, isso significa que mais de uma conexão é aberta/utilizada para as diversas requisições feitas. Conexões não persistentes são pouco menos vantajosas que as persistentes, pois levam uma fração maior de tempo de transmissão por causa das várias conexões que necessitam ser estabelecidas a cada pedido que é feito.

Caso a conexão entre os sistemas componentes do NTRIP seja perdida, os *sockets* - TCP são capazes de reconhecer perdas, e com isso, uma reconexão automática pode ser feita.

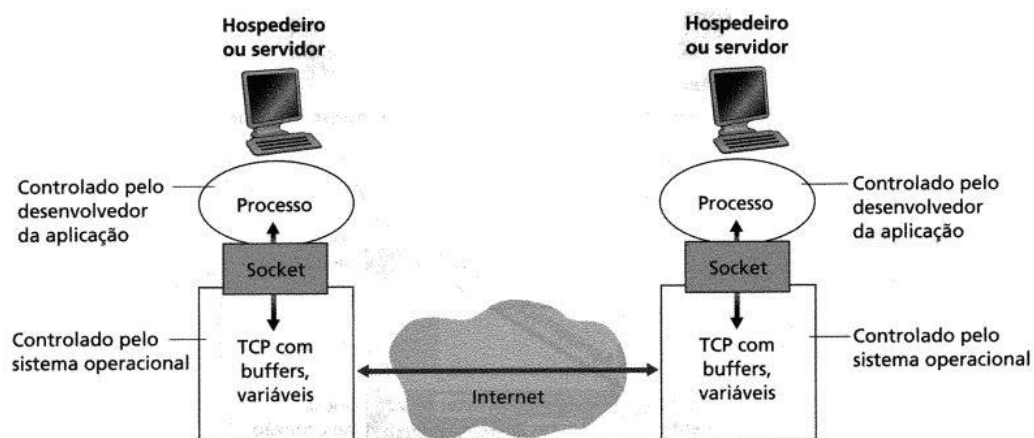


Figura 2: Sockets TCP e processos que os utilizam para comunicação (KUROSE J. F. E ROSS, 2005).

2.1.2.2 Servidores e Clientes HTTP

Como já foi dito anteriormente, o HTTP é implementado em dois programas distintos: o cliente e o servidor. Os servidores HTTP definem os servidores *Web*. Esses servidores contêm os diversos objetos *Web* que são solicitados pela parte cliente. Dentre os servidores populares da Web estão o Apache, como ilustrado na Figura 3, e o *Microsoft Internet Information Server* (KUROSE J. F. E ROSS, 2005). No NTRIP, o Caster atua como um servidor HTTP. Assim, é ele o servidor internet que mantém o fluxo de diferentes tipos de dados.

Já os clientes HTTP podem ser definidos como clientes *Web*. Esses clientes *Web* ou *browsers* são os responsáveis pela interação entre um usuário Internet e um servidor. São *browsers* populares o *Netscape Communicator* e o *Microsoft Internet Explorer* (KUROSE J. F. E ROSS, 2005).



Figura 3: Mensagens de requisição e respostas HTTP (KUROSE J. F. E ROSS, 2005).

2.1.2.3 Mensagens

Existem dois tipos de mensagens HTTP: as de requisição e as de resposta. Faremos uma breve descrição de cada uma dessas mensagens, mostrando ilustrações para uma melhor compreensão e explicando suas principais partes.



Figura 4: Formato geral de uma mensagem de requisição (KUROSE J. F. E ROSS, 2005).

2.1.2.4 Requisições HTTP

O formato geral de uma mensagem de requisição HTTP está ilustrado na Figura 4. Como é possível observar, uma mensagem deste tipo possui uma linha de requisição, linhas de cabeçalho, linha em branco e corpo da entidade. A seguir, detalhes sobre cada linha:

- **Método:** para esse primeiro campo da linha de requisição, existem diversos valores, variando de acordo com a versão do HTTP. Entre alguns deles estão o GET, POST e HEAD. A forma mais utilizada nas mensagens de requisição é o GET. Ele é empregado quando o browser deseja recuperar um objeto especificado no campo URL. O método POST é utilizado para enviar mensagens do cliente para o servidor, e o HEAD é para obtenção de informações sobre documentos ou objetos;
- **Sp (space):** espaço em branco para separar os campos da mensagem;
- **URL:** esse campo especifica o objeto que está sendo requisitado. A finalidade de uma URL é identificar o sistema final, onde o objeto que está sendo requisitado se localiza e o caminho para chegar até ele. Veremos mais adiante um exemplo de URL;
- **Versão:** esse campo especifica a versão do HTTP implementada na mensagem. A versão utilizada pelo NTRIP é o HTTP 1.1;
- **Cr(carriage return)e lf(line feed):** especifica que é o fim da linha e deve-se “pular” para a próxima linha;
- **Linhas de cabeçalho:** os campos de linha de cabeçalho especificam informações que devem ser passadas entre cliente e servidor, a fim de identificação. No cliente,

esse cabeçalho especifica suas configurações e o formato de seu documento. O cabeçalho pode conter mais de uma linha. Podemos tomar como exemplo a Figura 5 que nos mostra um cabeçalho com quatro linhas;

- **Corpo da entidade:** esse campo é utilizado em mensagens que contém o método POST. Para mensagens com método GET ele permanece vazio. Esse campo serve para capturar e enviar as informações que o usuário fornece, como em formulários, por exemplo.

Para uma compreensão mais clara de uma mensagem de requisição, observemos um exemplo:

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/4.0
Accept-language: fr
```

Figura 5: Típica mensagem de requisição HTTP (KUROSE J. F. E ROSS, 2005).

Examinando a Figura 5, podemos averiguar que a mensagem ali representada possui cinco linhas, sendo uma para requisição e as restantes para cabeçalho. Podemos observar na linha de requisição, que o método utilizado é o GET, a URL especificada é */somedir/page.html*, indicando o caminho onde pode-se encontrar o objeto no servidor, e por fim a versão do HTTP que é a versão 1.1. Observando agora o cabeçalho, na primeira linha temos *Host: www.someschool.edu*, especificando o servidor onde o objeto se encontra. Já na segunda linha temos a informação de que a conexão utilizada não é persistente: *Connection: close* especifica que a conexão deve ser fechada assim que o objeto for enviado. A linha de cabeçalho *User-agent : Mozilla/4.0* informa que o *browser* utilizado é do tipo Mozilla/4.0 e o cabeçalho *Accept-language: fr* especifica que o usuário tem preferência em receber o objeto na versão em francês, caso esse objeto não exista, a versão padrão é enviada.

2.1.2.5 Resposta HTTP

Já vimos como é a estrutura de uma mensagem de requisição, vamos agora analisar o formato geral de uma mensagem de resposta, ilustrada na Figura 6:

O formato geral da mensagem resposta, como mostra a Figura 6, aparentemente é



Figura 6: Formato geral de uma mensagem de resposta (KUROSE J. F. E ROSS, 2005).

semelhante ao formato da mensagem de requisição, porém existem algumas diferenças características. A seguir, uma explicação de cada campo dessa mensagem:

- **Versão:** esse campo especifica a versão do HTTP implementada na mensagem. É o mesmo campo da mensagem de requisição;
- **Sp (space):** espaço em branco para separar os campos da mensagem;
- **Código de estado:** esse campo indica qual foi o resultado obtido na requisição. Ele possui 3 dígitos e é associado com o campo frase, ambos complementam-se;
- **Frase:** esse campo detalha a mensagem de estado que foi referenciada no campo código de estado. Ele explica em formato texto o significado do código de três dígitos do campo anterior. Eis alguns códigos de estado e frases associadas comuns (KUROSE J. F. E ROSS, 2005):
 - *200 OK*: requisição bem-sucedida e a informação é entregue com a resposta;
 - *301 Moved Permanently*: objeto requisitado foi removido permanentemente; novo URL é especificado no cabeçalho *Location*: da mensagem de resposta. O software do cliente recuperará automaticamente o novo URL;
 - *400 Bad Request*: código genérico de erro que indica que a requisição não pode ser entendida pelo servidor;
 - *Not Found*: o documento requisitado não existe no servidor;
 - *505 HTTP Version Not Supported*: a versão do protocolo HTTP requisitada não é suportada pelo servidor;

- **cr(carriage return)e lf(line feed):** especifica que é o fim da linha e deve-se “pular” para a próxima linha;
- **Linhas de cabeçalho:** os campos de linha de cabeçalho especificam informações que devem ser passadas entre cliente e servidor, a fins de identificação. No servidor, esse campo especifica suas configurações e informações que detalham a resposta que está sendo enviada;
- **Corpo da entidade:** esse campo é a parte que mais interessa na mensagem. É nele que se encontra o objeto solicitado.

Apresentamos a seguir, uma típica mensagem de resposta HTTP. Vamos tomar esse exemplo como sendo uma resposta para o pedido feito anteriormente.

```
HTTP/1.1 200 OK
Connection: close
Date: Thu,03 July 2003 12:00:15 GMT
Server: Apache/1.3.0(Unix)
Last-Modified: Sun, 5 May 2003 09:23:24 GMT
Content-Length: 6821
Content-Type: text/html

(data data data data ...)
```

Figura 7: Típica mensagem de resposta HTTP (KUROSE J. F. E ROSS, 2005).

Nessa Figura 7, notamos que há, como primeira linha, a linha de estado. Também verificamos que possui seis linhas de cabeçalho, e por fim, o corpo da entidade. Vamos agora observar cuidadosamente o que descrevem todas essas linhas. Na primeira linha, denominada linha de estado, temos que a versão utilizada é o HTTP 1.1, e temos, como código de estado e frase, a mensagem 200 OK que nos indica que houve sucesso na requisição e entrega da resposta. Seguindo para a próxima linha, temos a informação de que a conexão TCP será fechada após o envio da resposta(confirmando que a conexão é não persistente). Em seguida temos a data e hora que o servidor localizou o objeto solicitado, o extraiu e o anexou na mensagem para enviá-lo. Temos também como mais uma linha de cabeçalho, indicado em *Server*: o tipo de servidor *Web* utilizado, que é o Apache. *Last-Modified*: indica a data e a hora da ultima atualização ou criação do objeto. Visualizando a figura, observamos que a ultima modificação/criação do objeto foi dia 5 de maio de 2003, aproximadamente às 9:23 hrs. As duas ultimas linhas do cabeçalho informam detalhes do objeto:*Content-Length*: diz a quantidade de bytes do

objeto, que no caso é 6821 bytes; e *Content-Type*: indica que o objeto está no formato texto HTML. Após a linha em branco, chegamos no corpo da entidade, que é o objeto em si, representado por (data data ...).

2.1.3 Protocolos da camada de Transporte

Conforme a hierarquia de camadas, abaixo da camada de aplicação, está a camada de transporte. Essa camada provê, à um processo da camada de aplicação, serviços de transporte para as suas mensagens. Essas mensagens são convertidas em pacotes menores que recebem um cabeçalho de identificação e assim, são mais fáceis de serem emitidas. Para realizar esse serviço, a camada de transporte oferece dois protocolos distintos : o TCP (*Transmission Control Protocol*) e o UDP (*User Datagram Protocol*). A camada de aplicação pode escolher um desses protocolos de acordo com as funcionalidades que cada um oferece. Esses dois protocolos e seus serviços serão apresentados detalhadamente nas próximas seções.

2.1.3.1 Protocolo TCP

O TCP - Protocolo de Controle de Transmissão - oferece um serviço confiável de transferência de dados. Diz-se serviço confiável pois ele dá a garantia de que os dados transmitidos chegarão ao destino e sem erros. Ao escolher o TCP como o protocolo, a aplicação não necessita se preocupar com detalhes de entrega, tudo isso é feito pelo protocolo. Esse protocolo se favorece de alguns mecanismos para garantir toda essa confiabilidade e robustez : usa controle de fluxo, controle de congestionamento, temporizadores de reenvio de mensagens perdidas e transmissão *Full duplex*. Todos esses mecanismos serão detalhados na próxima sessão.

2.1.3.2 Mecanismos TCP

Os mecanismos de controle de fluxo e controle de congestionamento, a primeira vista, podem parecer similares quanto aos seus objetivos. Porém cada um é executado por razões distintas. Controle de fluxo é a capacidade do TCP de compatibilizar velocidades: compatibiliza a taxa à qual o remetente está enviando com a aquela à qual a aplicação receptora está lendo. Controle de congestionamento é uma limitação na taxa a qual enviam tráfego para sua conexão. Se um remetente TCP perceber que há pouco congestionamento no cominho entre ele e o destinatário, aumentará sua taxa de envio; se perceber

que há congestionamento ao longo do caminho, reduzirá sua taxa de envio (KUROSE J. F. E ROSS, 2005). O controle de congestionamento possui como alvo maior a rede.

Esse protocolo, através de reconhecimentos, é capaz de perceber que pacotes foram recebidos corretamente e pode retransmitir os pacotes que foram corrompidos ou perdidos na transmissão. Os reconhecimentos podem ser feitos através de um campo presente em sua estrutura, como será estudado mais adiante.

Uma transmissão *Full duplex* é análoga a uma rodovia de duas vias: uma de “ida” e outra de “volta” em que os automóveis podem transitar nessa rodovia, nos dois sentidos, ao mesmo tempo. Assim é o serviço *full-duplex*: se houver uma conexão TCP entre o processo A em um hospedeiro e o processo B em um outro hospedeiro, então os dados da camada de aplicação poderão fluir de A para B ao mesmo tempo em que os dados da camada de aplicação fluem de B para A (KUROSE J. F. E ROSS, 2005).

2.1.3.3 Estrutura de segmento TCP

Em uma conexão TCP, todos os dados recebidos da camada de aplicação, recebem um cabeçalho TCP, constituindo assim, segmentos TCP. Esses segmentos são passados para a camada de rede e ficam sob responsabilidade do protocolo IP, que encapsula esses segmentos em datagramas e os envia na rede. Para entendermos melhor um segmento TCP, vamos então abordar sua estrutura. A Figura 8 mostra um segmento TCP com seu cabeçalho, e seu campo de dados. Vamos descrever então, o significado dos principais campos dessa estrutura (para mais detalhes, consultar (KUROSE J. F. E ROSS, 2005)):

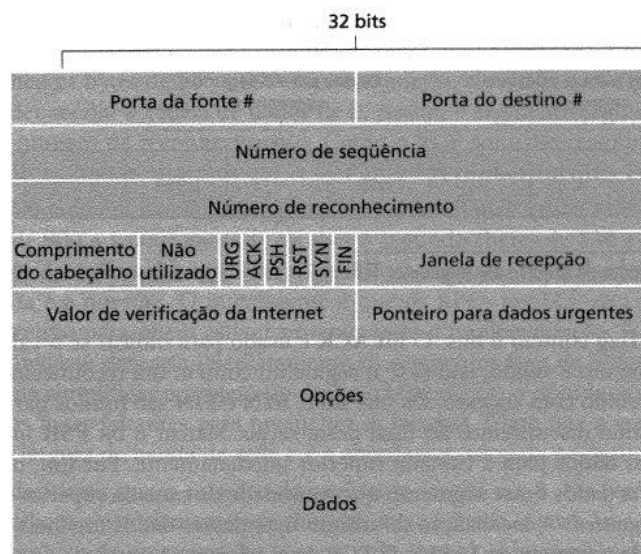


Figura 8: Estrutura de segmento TCP (KUROSE J. F. E ROSS, 2005).

- **Porta da fonte:** esse campo identifica o número da porta da entidade que enviou os dados para a rede, ou seja, é o número de identificação da porta do remetente;
- **Porta do destino:** o número contido neste campo identifica o número da porta do destinatário para a entrega dos dados. É através desse número que é feito um processo chamado demultiplexação. A demultiplexação pode ser comparada ao serviço do porteiro de um prédio residencial, que entrega em cada apartamento suas respectivas correspondências;
- **Número de sequência:** esse campo é utilizado para implementar a entrega confiável, indicando o primeiro byte do segmento que está sendo enviado. Para exemplificar, vamos supor que exista uma cadeia grande de bytes a ser transmitida. O TCP numera todos os *bytes* da cadeia, fragmenta toda essa cadeia em segmentos e os envia. Para identificar a ordem desses segmentos, é atribuído ao número de sequência, o número do primeiro *byte* desse segmento;
- **Número de reconhecimento:** esse campo revela ao destinatário, que esteja efetuando a troca de informações na mesma conexão TCP, o número da sequência de *bytes* pela qual ele está aguardando o recebimento. Como o TCP é *full-duplex* um hospedeiro A pode estar recebendo dados de um hospedeiro B enquanto envia dados ao hospedeiro B. Suponha que o hospedeiro A tenha recebido do hospedeiro B todos os bytes numerados de 0 a 535 e também que esteja prestes a enviar um segmento ao hospedeiro B. O hospedeiro A está esperando pelo byte 536 e por todos os *bytes* subsequentes da corrente de dados do hospedeiro B. Assim, ele coloca o número 536 no campo de número de reconhecimento do segmento que envia para o hospedeiro B;
- **Comprimento do cabeçalho:** esse campo especifica o comprimento do cabeçalho TCP em palavras de 32 bits ;
- **Flags:** o campo de flags possui 6 bits, um para cada flag. O bit ACK é usado para indicar se o segmento contém um reconhecimento para um segmento que foi recebido com sucesso. Os *bits* RST, SYN e FYN são usados para estabelecer e encerrar a conexão. O *bit* PSH indica que o destinatário deve passar os dados para a camada superior imediatamente e o *bit* URG, é usado para mostrar que há dados nesse segmento que a entidade da camada superior ao lado remetente marcou como 'urgentes' ;

- **Janela de recepção:** esse campo é utilizado para referenciar a quantidade de *bytes* que um destinatário pode receber sem que a demanda de dados afete seu desempenho, fazendo com que ocorram perdas. Assim, pode-se dizer que esse campo está voltado para o controle de fluxo do TCP;
- **Dados :** esse campo é destinado aos dados da aplicação.

2.1.3.4 Serviço Orientado à Conexão

O TCP oferece um serviço muito característico: o serviço orientado à conexão. Esse serviço faz com que uma “apresentação” inicial ocorra entre as entidades que desejam trocar informações. Essa apresentação é muito importante na transmissão pois permite que essas entidades possam se preparar para receber toda a informação que será transmitida. Nessa preparação, há alocações de variáveis e *buffers* que são utilizados durante toda a comunicação. Reveja a Figura 2, os *buffers* e as variáveis ilustradas foram alocadas na apresentação inicial.

Por fim, pode-se notar que uma aplicação que opta pelo TCP deseja toda segurança e garantia de seus dados.

2.1.3.5 Protocolo UDP

O UDP - Protocolo de Datagrama de Usuário - é um protocolo simples, sendo essa uma das diferenças do TCP, que é um protocolo complexo. O UDP não oferece a garantia da entrega confiável como o TCP e também não oferece um serviço orientado à conexão. Todavia, o UDP apresenta uma conexão mais rápida por não realizar verificações de entrega como o TCP. Esse protocolo pode implementar verificação de erros e faz uma tentativa de melhor esforço para entregar os dados. Não são muitas as características, porém uma aplicação sobre UDP pode implementar outras funcionalidades adicionais que sejam necessárias além dos serviços oferecidos pelo protocolo. Pacotes de dados podem se perder pelo caminho ou corromper-se, por isso, o UDP é ideal para aplicações de admitam corrompimento ou perdas do seu conteúdo. O protocolo é comumente usado para aplicações de multimídia, como telefone por internet, videoconferência em tempo real e recepção de áudio e vídeo armazenados, e todas essas aplicações podem tolerar uma pequena quantidade de perda de pacotes, de modo que a transferência confiável de dados não é absolutamente crítica para o sucesso da aplicação (KUROSE J. F. E ROSS, 2005).

2.1.3.6 Estrutura de segmento do UDP

O UDP possui quatro campos de cabeçalho somente. A Figura 9 mostra os quatro campos e também um campo de dados, no qual se encontram os dados provenientes da aplicação. Observando detalhadamente, percebe-se que cada campo do cabeçalho possui 16 *bits*, ou seja, 2 *bytes*, e todo o campo de dados possui 32 *bits* ou 4 *bytes*. Os detalhes de cada campo são dados a seguir (KUROSE J. F. E ROSS, 2005):

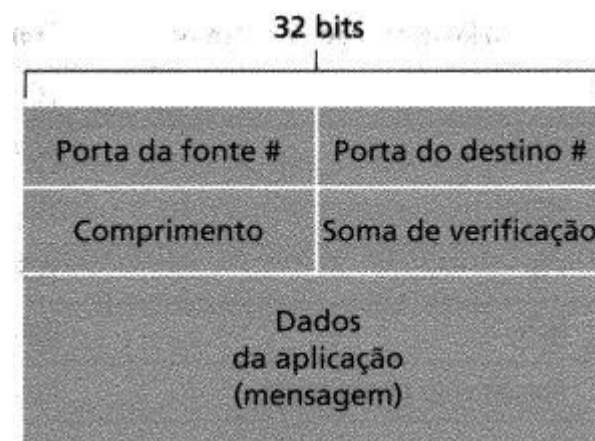


Figura 9: Estrutura de segmento do UDP (KUROSE J. F. E ROSS, 2005).

- **Porta da fonte:** esse campo identifica o número da porta do remetente da mensagem, ou seja, identifica quem enviou os dados. É análogo ao remetente colocado no envelope de uma carta;
- **Porta do destino:** o número da porta de destino, ao qual os dados devem ser entregues, é colocado nesse campo. É através desse número que o UDP direciona os dados ao destinatário, no processo de demultiplexação, semelhante ao TCP;
- **Comprimento:** esse campo especifica o comprimento do segmento UDP, incluindo o cabeçalho, em *bytes*;
- **Soma de verificação:** a soma é usada pelo hospedeiro receptor para verificar se foram introduzidos erros no segmento. O UDP no lado remetente realiza o complemento de 1 da soma de todas as palavras de 16 *bits* do segmento levando em conta o “vai um” em toda soma. Esse resultado é colocado no campo de soma da verificação no segmento UDP;
- **Dados da aplicação:** esse campo é destinado aos dados da aplicação que escolheu o UDP para realizar o transporte. Por exemplo, pra o DNS, o campo de

dados contém uma mensagem de consulta ou uma mensagem de resposta. Para uma aplicação de recepção de áudio, parcelas de áudio preenchem o campo de dados.

Pode-se notar que, embora o UDP tenha limitações características, existem razões, em determinadas aplicações, pelas quais ele é ideal para a utilização. À primeira vista, o UDP pode não aparentar ser o protocolo mais eficiente para atuar no transporte de dados, mas analisando detalhadamente, vemos que é possível implementar uma transferência confiável dos dados usando esse protocolo, e também vemos que há diversos motivos que levam à sua escolha.

2.2 NTRIP

Nesta Seção, é apresentado o protocolo NTRIP, descrevendo brevemente seu conceito e seus componentes. Será dada ênfase a sua estrutura, aos seus três componentes, e a sua atuação sob envio de dados. Também será apresentado de forma sucinta o conceito dos principais protocolos envolvidos, que atuam junto ao NTRIP, e como é feita a interação entre eles.

2.2.1 Protocolo NTRIP

O NTRIP foi desenvolvido pela Agencia de Cartografia e Geodésia Alemã (BKG - *Bundensamt für Katrographie und Geodäsie*), no Projeto Piloto EUREF-IP (*European Real-Time GNSS Pilot Project*) a fim de transmitir correções diferenciais (correções das coordenadas) dos dados GNSS pela Internet. O protocolo NTRIP é capaz de transmitir dados com formatos variados, mas o padrão de transmissão é o formato RTCM (*Radio Technical Comission for Maritime Services*).

Os diferentes fabricantes de receptores adotavam um formato específico para manipular os dados, ou seja, cada um adotava seu próprio modo de exibição dos dados, e isso se tornava uma grande dificuldade enfrentada pelos seus usuários. Sendo assim, só era possível trabalhar em RTK (*Real Time Kinematic*), se os equipamentos na estação de referência e móvel fossem do mesmo fabricante (COSTA et al., 2008). Para resolver essa situação, foi feito um acordo de padronização da transmissão de dados, adotando então, o RTCM como o formato padrão para os dados. O NTRIP veio com o intuito fornecer o fluxo de dados no formato RTCM em tempo real via internet. A seguir, uma breve descrição de algumas de suas características:

- É um protocolo sem estado (*stateless*), ou seja, após o envio de requisição e resposta, a conexão entre cliente e servidor é encerrada, não havendo armazenamento de estado de conexões anteriores;
- Consiste em um pequeno subconjunto de funções HTTP;
- Não limitado à tipos de dados específicos, mas projetado como protocolo genérico (RTCM, 2009);
- Suporta tanto usuários móveis quanto usuários estáticos (RTCM, 2009);
- Fácil de executar com recursos limitados do cliente e da plataforma de servidores;
- O usuário não necessita estar em contato direto com as estações de referência (estações com coordenadas conhecidas responsáveis pelo envio dos dados);
- Têm a capacidade de aceitar uma grande quantidade de usuários simultaneamente (COSTA et al., 2008);
- Está habilitado a fornecer o fluxo de dados através de qualquer rede móvel TCP/IP (*Transfer Control Protocol / Internet Protocol*) (COSTA et al., 2008);

2.2.2 Componentes do NTRIP

NTRIP é formado por três componentes de software : NtripServer, NtripCaster e NtripClient. Esses três componentes interagem da seguinte maneira: primeiramente, o NtripServer recebe dados de um receptor de dados GNSS(NtripSource) e os envia para o NtripCaster. O NtripCaster por sua vez, é responsável por disponibilizar os dados na Internet, de onde o NtripClient pode coletá-los. Para coletar os dados na Internet, o NtripClient necessita da autorização do NtripCaster, caso essa autorização seja concedida, o NtripClient passa então a receber os dados.

2.2.3 Ntrip Server

NtripServer é o componente responsável por distribuir os dados GNSS para o NtripCaster . Ele recebe os dados de uma fonte NtripSource e cria um canal de transferência para esses dados (como ilustrado na Figura 10). A aplicação mais simples NtripServer pode ser caracterizada como um componente de software instalado em um computador, implementado na linguagem C, que emite dados recebidos de algum receptor de dados GNSS. Um exemplo de um aplicativo NtripServer é o NtripServerCMD.

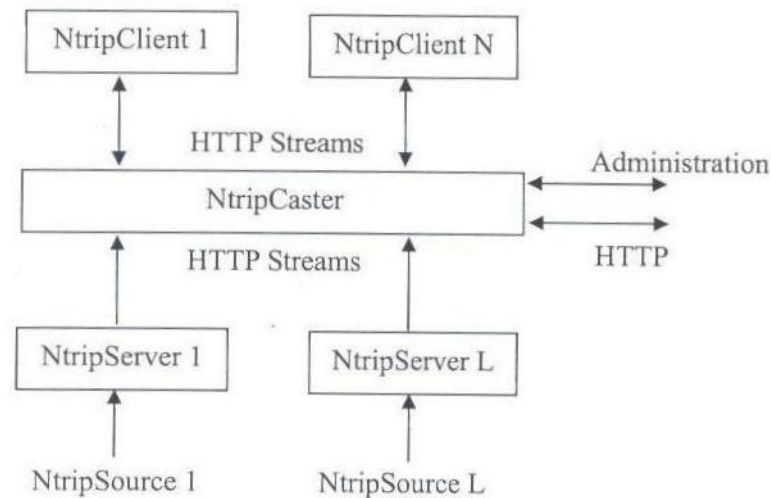


Figura 10: Estrutura e relacionamento dos componentes do NTRIP (RTCM, 2009).

Para transmitir os dados, o NtripServer precisa ser aceito pelo NtripCaster. Se for aceito, então é permitido a ele transferir os dados como identificações e parâmetros da fonte para o Caster.

2.2.3.1 Janelas de ajustes

No ambiente de transmissão, existem janelas de ajustes inclusas no sistema a fins de eficiência de operação. Assim, antes de começar a transmissão, são ajustadas algumas configurações tais como, configurações do fluxo de *bytes* a ser transmitido e da saída de dados para o NtripCaster, a senha e o *Mount point* fornecidos pelo administrador do NtripCaster, como detalhado na Figura 11.

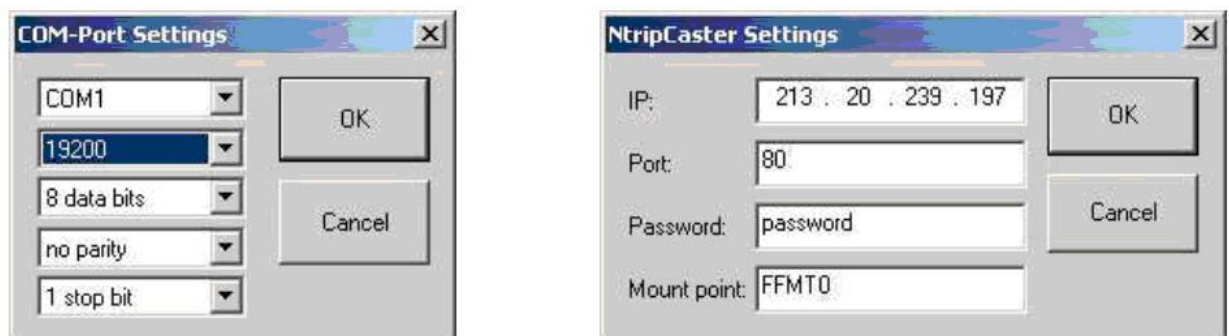


Figura 11: Janelas de ajustes no NtripServer (DETTMERING; WAESE; WEBER, 2006).

Também é possível editar configurações para casos de perda de conexão. Esta configuração é feita ativando-se a opção de conexão automática (*Use Autostart*) ilustrada na Figura 12, ao NtripCaster.

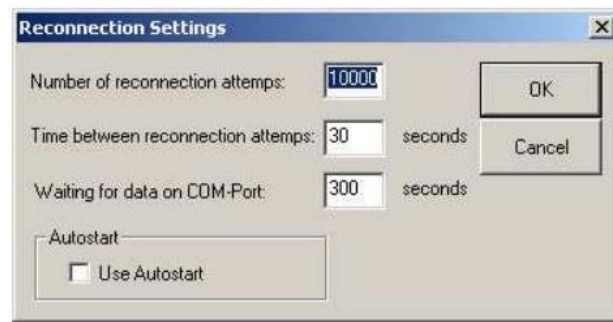


Figura 12: Janela de ajuste para conexão automática (DETTMERING; WAESE; WEBER, 2006).

2.2.3.2 Comunicação NtripServer para NtripCaster

Vimos na seção do protocolo HTTP suas mensagens de requisição e resposta. Vamos estudar agora uma mensagem de comunicação entre o NtripServer e o NtripCaster. A mensagem de pedido é semelhante à uma mensagem de requisição geral HTTP, iniciada pelo NtripServer (lembrando que NtripServer atua como um HTTP cliente), como mostra a Figura 13:

```
POST /ExampleMountpoint HTTP/1.1<CL><LF>
Host: ntrip.example.com<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
Autorization: Basic bnRyaXA6c2VjcmV0<CL><LF>
User - Agent: NTRIP ExampleServer/2.0<CL><LF>
Connection: close<CL><LF>
<CL><LF>
```

Figura 13: Mensagem de requisição NtripServer para NtripCaster (RTCM, 2009).

Como se pode observar na Figura 13, o método utilizado na linha de requisição é o método POST, indicando que o Server quer passar uma informação para o Caster. Também é possível verificar a versão utilizada do formato HTTP que é a versão 1.1, como já discutido nas sessões anteriores. A linha de cabeçalho *Host* indica o servidor: *ntrip.example.com*. *Ntrip-Version*: indica a versão do Ntrip utilizado, que é a 2.0. *Autorization*: é o controle de acesso ao *NtripCaster*, constituindo-se em login e senha secretos. O cabeçalho *User-Agent*: está indicando que o tipo de *browser* é *ExampleServer/2.0* e por fim temos a informação de que a conexão deve ser fechada após o envio dos dados - *Connection:close*.

A Figura 14 detalha a mensagem de resposta enviada ao NtripServer pelo NtripCaster, decorrente da requisição feita. Na primeira linha dessa mensagem, 200 OK indica que a

```

HTTP/1.1 200 OK<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
Server: NTRIP ExampleCaster/2.0<CL><LF>
Date: Tue, 01 Jan 2008 14:08:15 GMT<CL><LF>
Connection: close<CL><LF>
<CL><LF>

```

Figura 14: Mensagem de resposta NtripCaster para NtripServer (RTCM, 2009).

requisição feita pelo Server foi bem sucedida. O cabeçalho *Server: NTRIP ExampleCaster/2.0* indica que o servidor utilizado para gerar a mensagem é o *ExampleCaster/2.0*. *Date:* referencia a hora e a data que a resposta foi criada e enviada. A ultima linha, *Connection:* é utilizada para que o Server saiba que a conexão foi devidamente fechada após o envio da resposta. Como é possível observar, essa mensagem não contém uma entidade a ser enviada, já que a informação foi passada do NtripServer para o Caster e nada é preciso ser enviado pelo Caster, além da resposta de confirmação.

2.2.4 Ntrip Caster

NtripCaster faz a distribuição dos dados RTCM pela Internet, ou seja, disponibiliza os dados recebidos pelo NtripServer na internet, de onde os seus usuários conectados podem acessá-los. O NTRIP é capaz de suportar uma grande quantidade de usuários simultaneamente sem que seu desempenho seja afetado. Com isso, os usuários podem acessar, paralelamente, os dados por ele disponibilizados através de *login* e senha de acesso devidamente controlados. O Caster faz a verificação de registros dos usuários que desejam conectar-se a ele para poder autorizá-los ou não. Essa verificação é feita tanto nos pedidos do NtripServer quanto nos pedidos do NtripClient. Esse componente opera com mensagens de pedidos e respostas HTTP/RTSP (*Real Time Streaming Protocol*), e possui uma tabela conhecida como *sourcetable*, ilustrada na Figura 15, na qual estão indicadas cada estação de referência que solicita essas mensagens à ele. Essas estações de referência são identificadas como *mountpoints* e todas as suas configurações estão também descritas na tabela (*sourcetable*).

2.2.5 Ntrip Client

NtripClient é o componente que é executado nos dispositivos móveis ou não, como um celular ou PDA, responsável por acessar os dados na internet. A Figura 16 mostra um tipo de aparelho móvel que roda o NtripClient. Os dados que são distribuídos pelo Caster,

```

SOURCETABLE 200 OK
Server : NTRIP Caster 1.5.5/1.0
Date : 23/Jan/2004:08:54:59 UTC

Content-Type: text/plain
Content-Length: 6999
CAS;129.217.182.51;80;EUREF;BKG;0;DEU;51.5;7.5;http://igs.ifag.de/index_ntrip_cast.htm
CAS;62.159.109.248;8080;Trimble GPSNet;Trimble Terrasat;1;DEU;48.03;11.72;http://www.virtualrtk.com
NET;EUREF;EUREF;B;N;http://www.epncb.oma.be/euref_IP;http://www.epncb.oma.be/euref_IP;http://www.epncb.oma.be/projects/euref_IP/NtripRegister.html;none
NET;IGS;BKG;B;N;http://igsceb.jpl.nasa.gov;none;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;GREF;BKG;B;N;http://igs.ifag.de/gref_realtime.htm;http://igs.ifag.de/root_ftp/misc/ntrip/streamlist.htm;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;ascos;Ruhrgas AG;B;N;http://www.ascos.de;none;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;SCIGN-OCRT;SOPAC;CSRC;B;N;http://www.scign.org;none;ybock@ucsd.edu;none
NET;CORS;NGS;B;N;http://www.ngs.noaa.gov/CORS;http://igs.ifag.de/root_ftp/misc/ntrip/streamlist.htm;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;Misc;BKG;B;N;http://igs.ifag.de/root_ftp/misc/ntrip/streamlist.htm;none;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;EGNOS;BKG;B;N;http://igs.ifag.de/root_ftp/misc/ntrip/streamlist.htm;none;http://igs.ifag.de/root_ftp/software/NtripRegister.doc;none
NET;OmniSTAR;BKG;B;Y;http://www.omnistar.nl;http://www.omnistar.nl/graphs.asp;http://www.omnistar.nl/purchase.asp;Test and Evaluation only
STR;FFMJ2;Frankfurt;RTCM 2.1;1(1),3(19),16(59);0;GPS;GREF;DEU;50.12;8.68;0;1;GPSNet V2.10;none;N;N;560;Demo
STR;FFMJ1;Frankfurt;RTCM 2.2;3(19),16(59),18(1),19(1);2;GPS;GREF;DEU;50.09;8.66;0;0;GPSNet V1.9;none;N;N;2800;Demo
STR;FFMT0;Frankfurt;RTCM 2.0;1(3),2(90),3(90),16(90);0;GPS;Misc;DEU;50.09;8.66;0;0;Trimble 4000SSI;none;N;N;190;ALF Demo
STR;CONZO;Concepcion-TIGO;RAW;Compact(1);2;GPS+GLO;IGS;CHL;-36.84;286.98;0;0;Javad Legacy E;none;B;N;3600;none
STR;FFMJ0;Frankfurt;RAW;Compact(1);2;GPS+GLO;IGS;DEU;50.09;8.66;0;0;Javad Legacy E;none;N;N;3600;Demo
STR;LEIJ0;Leipzig;RAW;Compact(1);2;GPS+GLO;IGS;DEU;51.33;12.37;0;0;Javad Legacy E;none;B;N;3600;none
STR;LDBG0;Lindenberg;RAW;Compact(1);2;GPS+GLO;IGS;DEU;52.21;14.12;0;0;Javad Legacy E;none;B;N;3600;none
STR;MOXA0;Moxa;RAW;Compact(1);2;GPS+GLO;IGS;DEU;50.64;11.61;0;0;Javad Legacy E;none;B;N;3600;none
STR;MUEJ0;Muenchen;RAW;Compact(1);2;GPS+GLO;IGS;DEU;48.13;11.57;0;0;Javad Legacy E;none;B;N;3600;none
STR;WARN0;Warnemuende;RAW;Compact(1);2;GPS+GLO;IGS;DEU;54.17;12.10;0;0;Javad Legacy E;none;B;N;3600;none
STR;BARC0;Barcelona;RTCM 2.0;1(1),3(3),16(90);0;GPS;Misc;ESP;41.37;2.16;0;0;Novatel OEM3;none;B;N;230;ICC, Geod. Dep.
STR;BOCH0;Bochum;RTCM 2.1;1(1),3,16,18(1),19(1);2;GPS;Misc;DEU;50.45;7.27;0;0;none;none;B;N;13000;FH Bochum
STR;BRUS0;Brussels;RTCM 2.0;1(1),3(60),16;0;GPS;Misc;BEL;50.80;4.36;0;0;Ashtech UZ-12;none;B;N;500;ROB

```

Figura 15: Exemplo de Sourcetable (RTCM, 2005).

são utilizados pelo Client. Para ter acesso à esses dados, primeiramente é necessário obter uma autorização de conexão, que é concedida pelo NtripCaster. Assim sendo, se a autorização for concedida, ele passará a receber os dados disponibilizados. No NtripClient, o usuário seleciona um dos *mountpoints* existentes na tabela(*sourcetable*) do NtripCaster, escolhendo a fonte de onde ele deseja receber as informações. Será mostrado mais adiante como é feito o pedido da tabela para o NtripCaster.

2.2.5.1 Comunicação NtripClient para NtripCaster no NTRIP

A comunicação entre o NtripClient e o NtripCaster é feita em partes. Primeiramente, o Client solicita ao Caster as informações sobre os *mountpoints* disponíveis, como é ilustrado na Figura 17.

Essa mensagem de requisição é semelhante a mensagem de requisição do NtripServer, diferenciando-se pelo método GET, indicador de que o Client está fazendo uma requisição, e também pelo fato de não haver cabeçalho de autorização(não é necessário). Após o envio da mensagem de requisição da tabela, o NtripCaster envia a resposta com a correspondente *Sourcetable* (Figura 18):

Examinando detalhadamente a Figura 18, observamos que a mensagem ilustrada não diferencia-se de uma mensagem de resposta NTRIP, mas há uma linha de cabeçalho nova, que não foi abordada até agora - *NTRIP-Flags*: é uma informação utilizada somente em



Figura 16: Dispositivo, que roda o componente NtripClient, mostrando em tela alguns *mountPoints* de uma *Sourcetable* (GMBH, 2010).

```
GET / HTTP/1.1<CR><LF>
Host: ntrip.example.com<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
User - Agent: NTRIP ExampleClient/2.0<CR><LF>
Connection: close<CL><LF>
<CL><LF>
```

Figura 17: Mensagem de requisição da Tabela(*Sourcetable*) para NtripCaster (RTCM, 2009).

```
HTTP/1.1 200 OK<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
Ntrip-Flags: st_filter,st_auth,st_match,st_strich,rtsp<CL><FL>
Server: NTRIP ExampleCaster/2.0<CL><LF>
Date: Tue, 01 Jan 2008 14:08:15 GMT<CL><LF>
Connection: close<CL><LF>
Content - Type:gnss/sourcetable<CL><FL>
Content- Length: 1234<CL><FL>
<CL><FL>
sourcetable data
```

Figura 18: Mensagem de resposta com a Tabela(*Sourcetable*) para NtripClient (RTCM, 2009).

mensagens do Caster, e especifica as capacidades dos programas NTRIP, e os elementos desta linha especificam uma característica que é suportada na aplicação. *Content - Type*: identifica que o objeto sendo enviado (*sourceable data*) possui 1234 bytes. No mais, os detalhes e significados de todas as outras linhas já foram abordadas e estudadas. Após selecionar um *mountpoint* do qual quer receber os dados, o NtripClient passa a solicitar os dados ao Caster de acordo com esse *mountpoint* escolhido. A Figura 19 mostra um pedido de requisição de dados.

```
GET /ExampleMountpoint HTTP/1.1<CR><LF>
Host: ntrip.example.com<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
User - Agent: NTRIP ExampleClient/2.0<CR><LF>
Authorization: Basic bnRyaXa6c2VjcmV0<CR><LF>
Connection: close<CL><LF>
<CL><LF>
```

Figura 19: Mensagem de requisição NtripClient para NtripCaster (RTCM, 2009).

A mensagem de requisição acima (Figura 19) mostra o método GET sendo utilizado para requisição dos dados, indicando o endereço do *mountpoint* como endereço do sistema que disponibiliza o objeto solicitado. Há no cabeçalho, a linha *Authotization*: confirmando que login e senha são utilizados no Client para o controle de acesso feito pelo Caster. Também há a referência de que a conexão seja fechada após o envio dos dados.

Por fim, segue os detalhes da mensagem de resposta enviada pelo Caster, após autorizar o Client e confirmar o sucesso da requisição (ver Figura 20):

```
HTTP/1.1 200 OK<CL><LF>
Ntrip - Version: Ntrip/2.0<CL><LF>
Server: NTRIP ExampleCaster/2.0<CL><LF>
Date: Tue, 01 Jan 2008 14:08:15 GMT<CL><LF>
Cache-Control: no-store, no-cache, max-age=0<CL><LF>
Pragma: no-cache<CL><LF>
Connection: close<CL><LF>
Content - Type: gnss/ data <CL><LF>
<CL><FL>
data
```

Figura 20: Mensagem de resposta NtripCaster para NtripClient (RTCM, 2009).

A novidade da mensagem de resposta ao NtripClient, observada na ilustração acima, é a adição de duas novas linhas ao cabeçalho: *Cache-Control* e *Pragma*. Essas duas informações adicionais são recomendadas para mensagens no Caster, opcionais para o

Server e não utilizadas em mensagens Client, e possuem significado semelhante: indicam aos servidores *proxy* para desabilitar o cache. Finalizando, como corpo de entidade, os dados requisitados pelo Client.

2.3 RTCM

O padrão RTCM foi mencionado na Seção 3.2, e agora será discutido mais detalhadamente. É válido ressaltar o estudo desse capítulo pois os dados enviados pelo NTRIP encontram-se no formato RTCM e posteriormente, nesse projeto, são convertidos e gravados no formato de arquivo RINEX. Serão apresentados neste capítulo, o formato RTCM e suas principais versões e uma descrição sobre o formato RINEX.

2.4 Padrão RTCM

A RTCM - *Radio Technical Commission for Maritime Services* - é uma organização americana sem fins lucrativos. Objetivando melhorar a transmissão de dados GNSS, essa organização fundou um comitê para estabelecer os padrões e regras de transmissão (COSTA et al., 2008). Assim, nasceu, então, o padrão RCTM. A primeira versão foi o RTCM 1.0 e posteriormente novas versões foram surgindo, como a versão 2.1, 2.2 e 2.3. As versões mais recentes desse padrão permitem que diversas técnicas de posicionamento e aplicações sejam feitas, sendo esse um dos principais diferenciais das versões antigas. A versão mais recente é o RTCM 3.1. Nas próximas seções serão mostradas as principais versões e a estrutura de suas mensagens.

2.4.1 RTCM 2.3

A versão 2.3 possibilita a transmissão de um número limitado de mensagens, e suporta uma única estação de referencia, por não possibilitar um grande número de mensagens. Existem diversos tipos de mensagens RTCM 2.3, que são empregadas para diferentes finalidades, e todas elas são identificadas por um número de reconhecimento. Observe os tipos na Figura 21.

Vamos agora analisar a estrutura de uma mensagem do tipo 1, que é utilizada em correções diferenciais GPS. Uma mensagem do tipo 1, possui como cabeçalho duas palavras - padrão, sendo 30 *bits* o tamanho de cada palavra, como ilustrado na Figura 22. Cada mensagem, ou quadro de mensagem pode ter a quantidade total de $n+2$ palavras, sendo

Tipo de Mensagem	Status Atual	Nome da Mensagem
1	Fixa	Correções Diferenciais GPS
2	Fixa	Correções Diferenciais GPS Delta
3	Fixa	Parâmetros da Estação de Referência GPS
5	Fixa	Saúde da Constelação GPS
7	Fixa	Almanaque dos Radiofaróis DGPS
14	Fixa	Tempo da Semana GPS

Figura 21: Tipos de mensagens do RTCM 2.3 (DALBELO, 2010).

n o número de palavras para os dados e 2 o número de palavras de cabeçalho. O número máximo de palavras é 33, logo, percebemos que é possível utilizar 31 palavras para encapsular dados, e 2 palavras para o cabeçalho de identificação.



Figura 22: Cabeçalho da mensagem RTCM 2.3 com suas duas primeiras palavras. Fonte:(DALBELO, 2010).

As características de cada campo das palavras são definidas a seguir:

- **Preâmbulo:** esse campo possui um código de 8 bits que indica qual é o sistema utilizado. Convencionalmente o sistema de posicionamento GPS é identificado com o código 01100110;
- **Tipo da Mensagem:** esse campo de 6 *bits* identifica o tipo da mensagem que está sendo enviada, com um desses três tipos: fixa, quando seu padrão não pode ser alterado; tentativa, quando seu padrão está sujeito a alterações; e reservada, que aceita novos parâmetros implementados;
- **Identificação da estação:** esse campo é preenchido opcionalmente, definido pelo emissor;

- **Paridade:** os bits colocados nesse campo, são utilizados para verificar se há igualdade entre os bits enviados e recebidos. Um algoritmo é utilizado, denominado Código de Hamming, para geração dos bits de paridade;
- **Contador Z modificado:** este contador referencia o instante de tempo para os parâmetros da mensagem, ou o início do próximo quadro de mensagem;
- **Nº de seqüência:** esse campo possui 3 bits e contém o número de seqüência necessário para organizar e sincronizar as mensagens;
- **Comprimento do quadro:** esse campo revela o número de palavras da mensagem, sendo o comprimento = $n+2$, como já dito anteriormente;
- **Saúde da Estação:** nesse campo é inserido um código indicando a situação em que se encontra a estação de referência. A Figura 23 mostra os possíveis códigos que podem ser inseridos nesse campo e seus respectivos significados. O fator de escala indicado (UDRE- *User Differential Range Error*) é utilizado para mensagens de correções diferenciais.

Código	Indicação
111	Estação de referência não está funcionando
110	Transmissão da estação de referência não monitorada
101	Fator de escala UDRE = 0,1
100	Fator de escala UDRE = 0,2
011	Fator de escala UDRE = 0,3
010	Fator de escala UDRE = 0,5
001	Fator de escala UDRE = 0,75
000	Fator de escala UDRE = 1

Figura 23: Indicador de saúde da estação (DALBELO, 2010).

2.4.2 RTCM 3.1

Como já mencionado anteriormente, a versão 3.1 é a mais atual, e versões mais recentes possibilitam que diversos tipos de técnicas e aplicações sejam adicionadas à elas. A versão 2.3 por exemplo, não possibilita a aplicação RTK em Rede, pois não suporta oficialmente aplicações que utilizam rede de estações de referência. Já as versões mais recentes não têm essa limitação.

O RTCM 3.1 foi desenvolvido para suportar transmissões de uma grande quantidade de informações em tempo real para múltiplas estações de referência. Suas mensagens são

capazes de dar suporte a operações GPS e GLONASS. Outra característica dessa versão é que ela permite transportar mensagens contendo informações em texto, e permite também transportar informações de órbitas de satélites GPS e GLONASS.

As mensagens dessa versão, não possuem um formato padrão como um todo. Cada tipo de mensagem possui seu conjunto de parâmetros, e é agrupada de acordo com suas operações ou serviços. É possível observar na Figura 24 os diferentes grupos dos quais uma mensagem pode pertencer, é possível notar também que a mensagem 1001 é a menor e mais simples, ou seja, contém o mínimo necessário de informações das mensagens do grupo Observações GPS, seguida pela mensagem 1002 que contém as informações da mensagem 1001 mais outras informações adicionais.

Nome do grupo	Nome do sub-grupo	Tipo da mensagem
Observações	GPS L1	1001
		1002
	GPS L1/L2	1003
		1004
	GLONASS L1	1009
		1010
	GLONASS L1/L2	1011
		1012
Coordenadas da estação		1005
		1006
Descrição da antena		1007
		1008
Correções RTK em rede	Dados auxiliares das estações da rede	1014
	Diferenças de correções ionosféricas	1015
	Diferenças de correções geométricas	1016
	Diferenças das correções ionosféricas e geométricas combinadas	1017
Inf. auxiliares	Parâmetros do sistema	1013
	Dados de efemérides dos satélites	1019
		1020
	String de texto	1029
Informações proprietárias		Atualmente designadas de 4088 a 4095

Figura 24: Grupo de Mensagens do RTCM 3. Traduzido de (RTCM, 2006).

A Figura 25 apresenta o cabeçalho das mensagens do tipo 1001-1004 que pertencem

ao grupo das mensagens que realizam o transporte de observações GPS. Os parâmetros da mensagem são ordenados em lista. Os tipos de dados *uint* (números inteiros positivos) e *bit* (números binários) são seguidos por um número que representa o tamanho do campo. A seguir, detalhes sobre os campos de dados:

Campo de dados	Tipo de dados	Nº de bits
Número da mensagem (ex: "1001" = 0011 1110 1001)	uint12	12
Identificação da estação de referência	uint12	12
Tempo da época GPS (ToW)	uint30	30
Mensagem de sincronismo GNSS	bit (1)	1
Nº de sinais de satélite GPS processados	uint5	5
Indicador de Suavização livre de divergência GPS	bit (1)	1
Intervalo de suavização GPS	bit (3)	3
Total		64

Figura 25: Cabeçalho das mensagens RTCM 1001-1004. Traduzido de (RTCM, 2006).

- **Número da mensagem:** campo que identifica a mensagem;
- **Identificação da estação de referência:** esse campo identifica uma estação de referência na aplicação;
- **Tempo da época GPS (ToW - *Time of Week*):** esse campo indica o número de segundos passados desde o início da semana GPS até o instante que é feita a observação da mensagem;
- **Mensagem de sincronismo GNSS:** nesse campo, um flag é responsável por indicar ao receptor se os dados podem ou não ser processados logo após o recebimento dessa mensagem. Isso porque, com o flag de valor 0, o campo indica que não há uma informação GNSS adicional para ser enviada, e o receptor já pode processar os dados; e com o flag de valor 1, o campo indica que há observações GNSS na próxima mensagem e que devem ser aguardadas;
- **Nº de sinais de satélite GPS processados :** indica o número de satélites que estão na mensagem;
- **Indicador de Suavização livre de divergência GPS:** se nesse campo o número presente for 1, a suavização livre de divergência é utilizada, se o número for 0, significa que a suavização não é utilizada. A suavização livre de divergência é feita através de um algoritmo de suavização;

- **Intervalo de suavização GPS:** referencia o intervalo no qual a pseudodistância é suavizada.

Como ja foi dito, cada tipo de mensagem possui um conjunto de parâmetros, e, sucessivamente, parametros adicionais são inclusos. Nota-se que, mensagens em um mesmo grupo contêm dados similares, cada qual contendo informações adicionais para melhorar seu desempenho.

2.4.3 Estrutura RTCM 3

Uma estrutura é utilizada para envio e recebimento de mensagens RTCM nas versões 3.0 e 3.1. As mensagens contendo toda a informação a ser transmitida são encapsuladas nessa estrutura. A estrutura possui cinco campos que podem ser observados na Figura 26:

Preâmbulo	Reservado	Tamanho da Mensagem	Mensagem de tamanho variável	CRC
8 bits	6 bits	10 bits	Tamanho variável, n^2 inteiro de bytes	24 bits
11010011	Não definido, estabelecido como 000000	Tamanho da mensagem em bytes	0 – 1023 bytes	QualComm CRC-24Q

Figura 26: Estrutura de uma mensagem do formato RTCM versão 3. Traduzido de (RTCM, 2006).

- **Preâmbulo:** esse campo possui 8 bits e é fixo;
- **Reservado:** os seis *bits* desse campo devem ser reservados e é estabelecido que para todas as mensagens são todos bits 0;
- **Tamanho da Mensagem:** referencia o tamanho total da mensagem;
- **Mensagem de tamanho variável:** campo onde se encontra a mensagem RTCM que pode ter seu tamanho variável;
- **CRC(Checação de Redundância Cíclica):** 24 bits de paridade são utilizados para o algoritmo QualComm CRC com permissão. Esse algoritmo faz a detecção de erros de *bits* que podem ser aleatórios, ou uma sequência de erros de *bits*;

2.5 Formato RINEX

O formato RINEX (*Receiver Independent Exchange Format*) foi desenvolvido pelo Instituto Astronômico da Universidade de Berna a fim de facilitar a interpretação e troca de informações entre diferentes receptores de dados GPS. Posteriormente foi adicionada a possibilidade de incluir nesse formato os dados advindos de outros sistemas de satélite como o GLONASS e SBAS (GURTNER; ESTEY, 2007).

Para a utilização e entendimento de dados (informações) gerados a partir dos diferentes tipos de receptores, esses dados são convertidos e escritos em um arquivo RINEX, um formato padrão de arquivos que pode ser interpretado e compreendido por diferentes programas de processamento GPS. O principal foco desse formato são as observáveis (ou grandezas), junto com informações relativas às estações, utilizadas pelos softwares de sistema de navegação: fase (medidas de fase da portadora), código (medida da pseudodistância), e o tempo de observação.

2.5.1 Especificação das Observáveis

Como já mencionado, três grandezas fundamentais são utilizadas e necessitam ser bem definidas: o tempo, código (medida da pseudodistância) e a fase.

- **Fase:** é a medida da fase da portadora que é dada em ciclos inteiros. Para esta grandeza são contados todos os ciclos completos de onda portadora, desde sua saída no satélite até a chegada no receptor;
- **Tempo:** a medida do tempo de uma observação é o momento em que o receptor recebeu os sinais (observações). É o mesmo tempo para as medidas de fase e código, e também para todos os satélites observados nesse mesmo momento. É expresso no Sistema de Tempo GPS e não no Sistema Universal de Tempo (GURTNER; ESTEY, 2007);
- **Pseudo- distancia:** é a medida da distancia entre a antena do receptor e a antena do satélite, levando em conta também os efeitos atmosféricos e erros do relógio do satélite e do receptor (GURTNER; ESTEY, 2007). A equação de pseudodistancia pode ser escrita da seguinte maneira :

$$PD = \text{distância geométrica} + c (\text{erro de relógio do receptor} - \text{erro de relógio satélite}) + \text{outros erros}$$

onde c é velocidade da luz(299792458 m/s). O resultado dessa equação é dado em metros.

- **Doppler:** esta é uma observável adicional(deslocamento Doppler) para informar caso haja variação na frequência do sinal, decorrente à movimentação(deslocamento) do satélite.

2.5.2 Versões

Existem várias versões disponíveis desse formato, sendo a versão 2.1, 2.11 e 3.0 as utilizadas por este projeto. O RINEX 2.11 consiste em 7 diferentes tipos de arquivos:

- Arquivo de Dados de Observação;
- Arquivo de Mensagem de Navegação;
- Arquivo de dados Meteorológicos;
- Arquivo de Mensagem de Navegação GLONASS;
- Arquivo de Mensagem de Navegação GEO;
- Arquivo de erro do relógio do Receptor e do Satélite;
- Arquivo de Dados Transmitidos pelo SBAS.

Já o RINEX 3.0 consiste em 3 tipos diferentes de arquivos ASCII:

- Arquivo de dados de Observação;
- Arquivo de Mensagens de Navegação;
- Arquivo de dados Meteorológicos.

Independente da versão, cada tipo de arquivo contém uma seção de cabeçalho, com informações sobre o arquivo, e uma seção de dados. O cabeçalho está no início do arquivo e nele estão descritos os detalhes sobre o interior do arquivo, como o tipo de observações que serão armazenadas pelo receptor e os sistemas de satélites que estão sendo observados. Existe uma rotulação ao final de cada linha contida na seção de cabeçalho, que descreve o seu significado. Esta rotulação é obrigatória e deve aparecer nos arquivos exatamente

```

GNSS OBSERVATION DATA FILE - EXAMPLE
-----|-----1|0|-----2|0|-----3|0|-----4|0|-----5|0|-----6|0|-----7|0|-----8|0-----

3.00          OBSERVATION DATA      M          RINEX VERSION / TYPE
G = GPS  R = GLONASS  E = GALILEO  S = GEO  M = MIXED COMMENT
XXRINEXO V9.9      AIUB              20060324 144333 UTC PGM / RUN BY / DATE
EXAMPLE OF A MIXED RINEX FILE VERSIOIN 3.00 COMMENT
The file contains L1 pseudorange and phase data of the COMMENT
geostationary AOR-E satellite (PRN 120 = S20) COMMENT
A 9080 MARKER NAME
9080.1.34 MARKER NUMBER
BILL SMITH OBSERVER / AGENCY
X1234A123 GEODETTIC 1.3.1 REC # / TYPE / VERS
G1234 ROVER ANT # / TYPE
4375274. 587466. 4589095. APPROX POSITION XYZ
.9030 .0000 .0000 ANTENNA: DELTA H/E/N
0 RCV CLOCK OFFS APPL
G 5 C1C L1W L2W C1W S2W SYS / # / OBS TYPES
R 2 C1C L1C SYS / # / OBS TYPES
E 2 L1B L5I SYS / # / OBS TYPES
S 2 C1C L1C SYS / # / OBS TYPES
18.000 INTERVAL
G APPL DCB xyz.uvw.abc//pub/dcb gps.dat SYS / DCBS APPLIED
DBHZ SIGNAL STRENGTH UNIT
2006 03 24 13 10 36.0000000 GPS TIME OF FIRST OBS
END OF HEADER

```

Figura 27: Cabeçalho de um arquivo de observação RINEX 3.0 (GURTNER; ESTEY, 2007).

como o padrão especifica, da coluna 61 a 80. Veja a Figura 27 que exemplifica um cabeçalho de arquivo Rinex versão 3.0.

Observando a Figura 27, é possível notar que na primeira linha está especificada a versão utilizada do RINEX : “3.00”, que o tipo do arquivo é o de Observação: “OBSERVATION DATA”, e que este arquivo contém dados de vários sistemas de satélites: “M” , como especificado ao final de cada linha. Também é possível notar, na terceira linha, a data e a hora que o arquivo foi criado, sendo a data 24/03/2006 e o horário 14:43:33 hrs.

2.5.3 Especificações dos arquivos RINEX

Cada arquivo de observação e cada arquivo de dados meteorológicos contém dados de uma estação. A partir da versão 2 também é possível a inclusão de dados de observação de mais de uma localização ocupada pelo receptor, no caso de um receptor se movendo ao longo de uma trajetória, em aplicações estático rápido ou cinemática. Porém, não é recomendável que seja inserido em um mesmo arquivo dados sobre mais de um receptor, mesmo que os dados não sobreponham-se no tempo (GURTNER; ESTEY, 2007).

Os arquivos de navegação contém informações sobre os satélites, como parâmetros orbitais que permitem calcular sua posição e velocidade. No GLONASS, as mensagens já vêm com posição, velocidade e aceleração para o satellite. Arquivos de navegação RINEX

3.0 podem apresentar em seu conteúdo mensagens de navegação de mais de um sistema de satélite (GURTNER; ESTEY, 2007), ou seja, podem conter mensagens de navegação GPS, GLONASS, GALILEO, ou SBAS.

2.6 J2ME - JAVA Micro Edition

Cada vez mais o ser humano encontra e tenta encontrar maneiras para facilitar e agilizar suas tarefas essenciais do dia a dia. Desta forma surgiram os *Fast Foods*, os *Deliverys*, que suprem a necessidade da alimentação rápida. Da mesma forma estão surgindo os *netbooks*, os *notebooks*, os *palmtops*, e os *tablets* que são ferramentas muito úteis de conciliação do trabalho, comunicação, e mobilidade, e que nada mais são do que tecnologias móveis. A tecnologia móvel surgiu como facilidade e transformou-se em necessidade. Os equipamentos eletrônicos para o consumidor modificam nossas vidas. Os telefones móveis nos permitem comunicar quando estamos longe de casa ou do escritório. Os PDA's nos permitem acessar *e-mails*, navegar na Internet e executar aplicativos de todas as formas e modalidades (MUCHOW, 2007). a seção será abordada de tecnologia móvel, apresentando as diferentes tecnologias que atuam sobre os dispositivos móveis. Também será explicada a plataforma J2ME (*Java 2 Micro Edition*), uma das plataformas Java existentes para o desenvolvimento dos aplicativos para os dispositivos.

2.6.1 Dispositivos Móveis

“Computação Móvel” está na moda. Operadoras de celular propagam marketing sobre suas aplicações móveis; os fabricantes de PDAs (*Personal Digital Assistant*) gastam milhões de dólares para mostrar ao mundo seu novo dispositivo móvel (JOHNSON, 2007). Sendo assim, o celular deixou de ser um aparelho eletrônico utilizado para transmitir, apenas, sinal de voz. As pessoas entenderam que além de ser um instrumento eficaz na comunicação verbal, ele também pode ser um instrumento bastante eficaz em soluções que estejam disponíveis para o usuário onde quer que ele se encontre. Seja como um minicentro de lazer, ou como uma ferramenta de trabalho, o celular tornou-se um objeto muito útil para o usuário. Hoje, ele é capaz de oferecer diversas atividades como tocar músicas, tirar fotos, gravar filmes e conectar-se a redes para acesso à internet, disponibilizando funcionalidades avançadas que podem ser otimizadas através de aplicativos executados em seu Sistema Operacional. Entram na lista de dispositivos com computação móvel, os *notebooks* ou *laptops*, os PDAs e os celulares. Cada um desses tipos de aparelhos, embora

possuam características semelhantes, como acesso comunicação sem fio, diferem na forma em que seus aplicativos são gerados, cada um de acordo com a linguagem suportada pelo aparelho (*hardware*), como será visto mais adiante.

2.6.2 Plataforma JAVA

O *Java Platform Micro Edition* é um dos ambientes Java direcionado para as tecnologias dos dispositivos com limitações de recursos, se comparados aos computadores de mesa. Esse ambiente é capaz de abranger um largo leque dos diferentes dispositivos e tecnologias como ilustrado na Figura 28.



Figura 28: Plataforma Java (ROMEIRO, 2005).

Discorrendo brevemente sobre a figura, podemos observar as quatro plataformas, ou ambientes Java, definidas em patamares diferentes, de acordo com a abrangência de cada uma.

O JEE ou *Java Enterprise Edition* é a maior plataforma e dedicada para o desenvolvimento de aplicações servidoras e de grande porte como sistemas corporativos. Possui uma vasta gama de bibliotecas que permitem o acesso a servidores, sistemas de *e-mail*, bancos de dados, e páginas *Web*. Sendo assim, temos as seguintes especificações incluídas em sua API : *JavaMail* para tratamento de sistemas de email, *JSP* e *Servlets* para desenvolvimento de páginas *Web* dinâmicas, e *JDBC* para manipulação de bancos de dados.

Por todas essas características é capaz de oferecer suporte a muitos usuários ao mesmo tempo.

O JSE ou *Java Standard Edition* é o ambiente voltado para desenvolvimento de aplicações para *Desktop*. Pelos diversos tipos de aplicativos e sistemas que esses computadores individuais necessitam, é a plataforma mais estudada e procurada pelos desenvolvedores.

O JME é o ambiente voltado para aplicações destinadas a pequenos dispositivos portáteis ou móveis. Essa tecnologia Java permitiu que muitas funcionalidades como jogos e agendas fossem incluídas nos aparelhos, trazendo muitos benefícios para as empresas desenvolvedoras. Sua biblioteca e configurações contém funções especialmente para o tratamento de suas limitações de memória e processamento.

O Java Card é a menor plataforma e tem como principal característica a segurança. Ela garante a privacidade de informações e ambiente seguro de execução para smart cards.

2.6.3 Arquitetura J2ME

Para permitir a utilização da mesma plataforma em diferentes dispositivos, a J2ME foi dividida em camadas (JOHNSON, 2007). A arquitetura do Java ME está dividida em três camadas: Máquina Virtual, Configuração e Perfil. Observe a Figura 29:

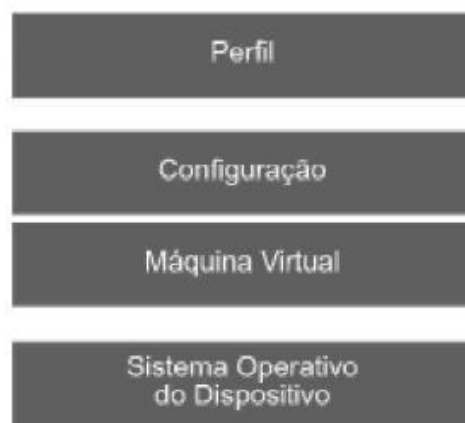


Figura 29: Camadas da arquitetura J2ME (MUCHOW, 2007).

A Máquina Virtual é a camada acima do Sistema Operacional e é nela que se encontram as restrições do dispositivo, ou seja, ela define quais são as restrições dos programas que executam no dispositivo. Nesta camada, são definidas duas máquinas virtuais apropriadas para os dispositivos com limitações de memória: a JVM (*Java Virtual Machine*) e

uma máquina virtual chamada de KVM (*Kilo Virtual Machine*), menor, mas que mantém as características da máquina virtual Java tradicional e utilizada para a configuração CLDC como será visto mais adiante.

A Configuração é a camada que define bibliotecas, classes e APIs, características da linguagem e alguns recursos da máquina virtual, estando assim, a camada de Configuração e Máquina virtual, vinculadas. Existem dois tipos de configuração definidos : CLDC e CDC.

- **CLDC (*Connected, Limited Device Configuration*)** : os dispositivos com essa configuração são bastante limitados quanto aos recursos de hardware. Nesta categoria se encontram dispositivos com processadores de 16 ou 32 bits e máxima de 512 kB de memória total. Possuem bibliotecas básicas e uma máquina virtual - KVM. Essa configuração foi projetada para aparelhos com baixas propriedades gráficas e baixo processamento, com conexões intermitentes de rede. Aqui , estão os celulares, os Pagers e Smartphones;
- **CDC (*Connected Device Configuration*)**: nessa configuração estão os aparelhos com mais capacidade de processamento, maiores propriedades gráficas e memória, e também com conectividade rápida de rede, diferente do CLDC. Dispositivos com essa configuração possuem, no mínimo, processadores com 32 bits e 2,5 MB de memória não volátil. Os aparelhos com essa configuração são alguns PDAs e sistemas de navegação para automóveis.

O Perfil é a camada que “prolonga” a Configuração. Essa camada define classes específicas, bibliotecas e funcionalidades que provêm usos específicos de um tipo particular de dispositivo. O perfil é o responsável pela interface com o usuário, ou seja, é a camada mais visível para os usuários e desenvolvedores. Para isso, define classes de interface para o desenvolvimento de aplicações que fazem essa interação.

Uma configuração é capaz de dar suporte a vários perfis, mas um perfil é sempre implementado para uma configuração específica. Por exemplo, um perfil definido para a configuração CLDC é o MIDP(*Mobile Information Device Profile*) como é possível observar na Figura 30, esse perfil é comumente utilizado e define APIs para tratamento de eventos, como ativar e desativar aplicações de dispositivos móveis com determinadas limitações(tela e memória) e também para suporte a protocolos de rede como o http e *sockets*(utilizados neste projeto). Outro perfil semelhante é o IMP, mais recente e praticamente um subconjunto de MIDP, para dispositivos com limitações maiores em interface

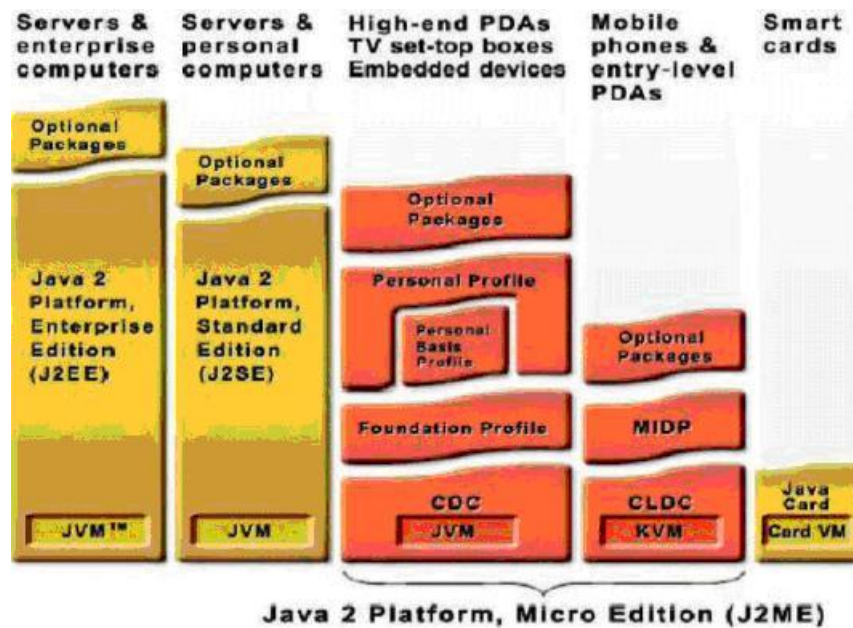


Figura 30: Ambientes da plataforma Java e os Perfis da J2ME. Adaptado de (JOHNSON, 2007).

gráfica. Os perfis que especificam a CDC são (Figura 30): *Foundation Profile* (FP), perfil básico, não oferece suporte à interface gráfica pois sua finalidade é para dispositivos como hardwares embutidos; *Personal Basic Profile* (PBP), perfil que inclui a interface gráfica mas sem compatibilidade total com a AWT (*Abstract Windowing Toolkit*); e *Personal Profile* (PP), mais completo e popular, possuindo todas as funcionalidades de PBP e FP e compatibilidade com a AWT.

Capítulo 3

Desenvolvimento e Testes

3.1 Considerações iniciais

Neste capítulo, é discutido o desenvolvimento do aplicativo proposto por este projeto. São mostrados todos os passos e configurações necessárias à implementação e também aos testes, desde a ferramenta escolhida, até o resultado final: os arquivos RINEX gerados. O capítulo encontra-se como segue:

- Na Seção 6.2 é descrito todo o processo de desenvolvimento;
- Na Seção 6.3 são mostrados os testes efetuados e o resultado.

3.2 Desenvolvimento

O aplicativo desenvolvido baseou-se em uma aplicação já existente para Desktop, desenvolvida em C, chamada `rtcm3torinex`. Essa aplicação encontra-se disponível para o download em: www.igs.bkg.bund.de/ntrip/download. O `rtcm3torinex` é um programa conversor em tempo real, de dados RTCM 3, transmitidos pelo NTRIP, para o formato RINEX (GMBH, 2009). Sendo assim, o aplicativo implementado teve como objetivo gerar arquivos RINEX através da plataforma J2ME.

3.2.1 Ferramentas utilizadas

Para a implementação foi utilizada a ferramenta Netbeans 6.8 IDE, no qual o código foi escrito e compilado. O Netbeans é um ambiente de desenvolvimento que auxilia

programadores/desenvolvedores na criação de aplicativos multiplataforma. A interface da IDE pode ser observada na Figura 31.

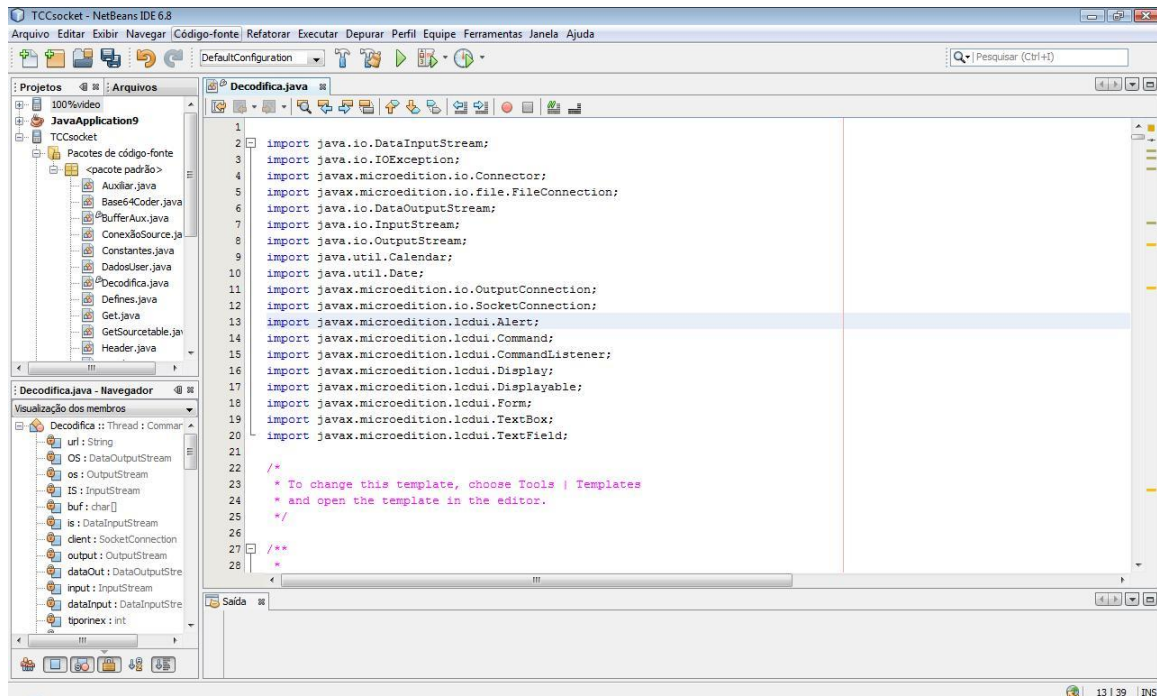


Figura 31: Netbeans IDE 6.8.

Ao iniciar um projeto JME, foi necessário efetuar algumas configurações antes da implementação, como o tipo de Perfil e Configuração desejados para o projeto. A descrição dessas configurações foram definidas na seção 5.4 e são elas:

- Configuração de dispositivo CLDC-1.1;
- Perfil do dispositivo MIDP-2.1;
- Plataforma do Emulador Java(TM) *Platform Micro Edition* SDK 3.0. A Java ME SDK 3.0 é uma ferramenta que integra diversas funcionalidades em um mesmo SDK. Ele integra CLDC, CDC e a tecnologia *Blu-ray Disc* Java (BD-J) em um único SDK (SUN, 2009). O download e mais informações sobre a ferramenta estão disponíveis em <http://www.oracle.com/technetwork/java/javame/javamobile/download/sdk/index.html>.

3.2.2 Arquitetura da Aplicação

3.2.2.1 MIDlets

A arquitetura do aplicativo está estruturada sobre a forma de um aplicativo MIDP. As aplicações do MIDP são implementações de uma classe abstrata chamada MIDlet (JOHNSON, 2007). Esta classe faz o gerenciamento do ciclo de vida de uma aplicação, que pode estar em três diferentes estados : ativo, pausado ou destruído. Observe a Figura 32 que mostra a vida de um MIDlet.

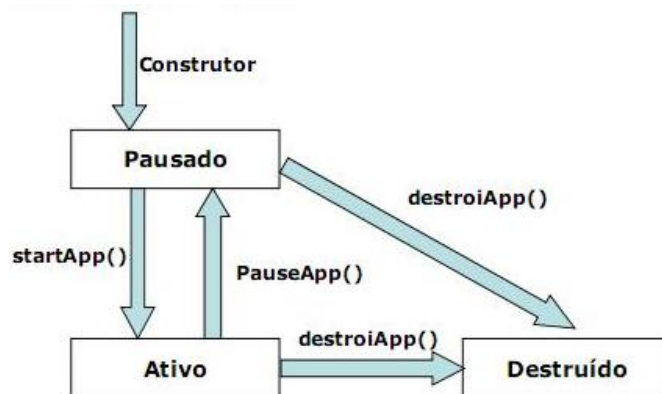


Figura 32: Ciclo de vida de um MIDlet (JOHNSON, 2007).

Como pode-se observar, para criar um MIDlet, esses três estados necessitam ser implementados em métodos. Cada método indica em qual estado de vida o aplicativo se encontra e são obrigatórios em todos MIDlets.

Após a chamada do construtor, o MIDlet permanece carregado, esperando por uma ação(estado Pausado). Quando a aplicação é iniciada pelo método `startApp()` o MIDlet passa então para o estado Ativo. O método `pauseApp()` faz com que a execução seja pausada e, o método `destroyApp()` termina a execução, logo, o MIDlet passa para o estado Destruído. Quando uma MIDlet tiver sido iniciada, ela pode alternar entre os dados de pausa e ativo qualquer número de vezes, durante seu ciclo de vida (MUCHOW, 2007).

3.2.2.2 Arquitetura

O aplicativo baseou-se na arquitetura cliente-servidor. Em uma arquitetura cliente-servidor há um hospedeiro sempre em funcionamento, denominado *servidor*, que atende a requisições de muitos outros hospedeiros, denominados *clientes*. Esses podem estar em funcionamento às vezes ou sempre (KUROSE J. F. E ROSS, 2005).

A Figura 33 mostra o diagrama das principais partes da aplicação, sendo “Conexão com o Servidor” o momento que a aplicação conecta-se ao servidor para efetuar a requisição os dados.

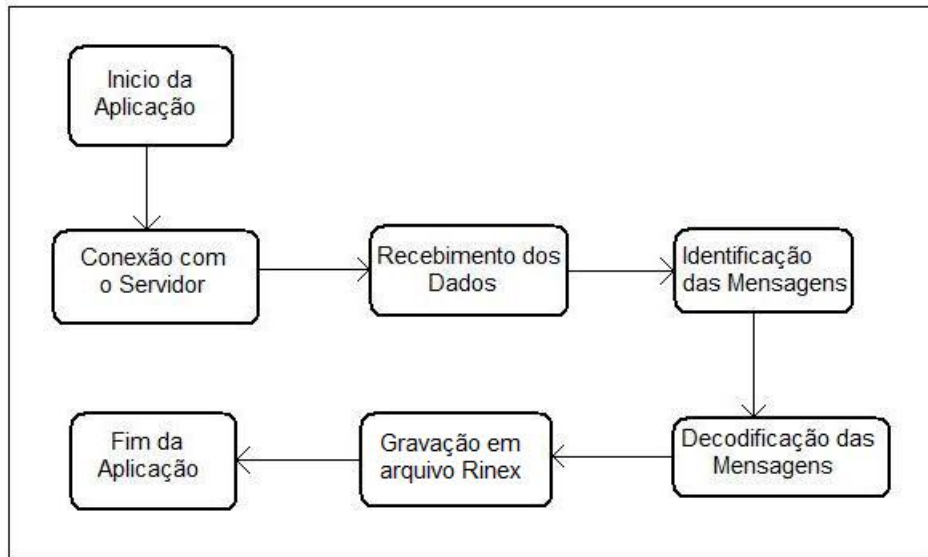


Figura 33: Diagrama das principais partes da aplicação.

Para criar um link de comunicação com o servidor, foram utilizados sockets. A implementação através de *socket* foi estabelecida utilizando as interfaces de conexão *Connection* e *SocketConnection* disponibilizadas pelo GFC(*Generic Connection Framework*), *framework* básico de conexão do J2ME. O código referente à conexão pode ser observado na Figura 34.

```

client = (SocketConnection)Connector.open(url);
client.setSocketOption(SocketConnection.DELAY, 0);
client.setSocketOption(SocketConnection.LINGER, 5);
client.setSocketOption(SocketConnection.KEEPALIVE, 0);
  
```

Figura 34: Conexão utilizando socket.

Na figura, *client* é uma instância de *SocketConnection*, *url* é a string de conexão e está no formato *protocolo:[alvo]/[parâmetros]*. A url utilizada para esta conexão foi *socket://<host>:<porta>*, sendo *socket* o protocolo que indica o tipo da conexão criada e, servidor e porta o endereço do servidor e porta ao qual o usuário deseja se conectar. As demais linhas implementadas indicam configurações setadas para o tratamento da conexão, como envio/recebimento de pequenos pacotes de dados, tempo de espera para o fechamento da conexão e validade da conexão e pacotes.

Após a conexão ser estabelecida, é montada uma mensagem de requisição, como vista na Figura 19(página 32), contendo os parâmetros setados pelo usuário. Essa mensagem é enviada ao servidor no qual foi estabelecida a conexão. Na Figura 35 está detalhada a implementação do envio da mensagem e do recebimento dos dados enviados pelo servidor.

```

for (int i = 0; i < buff.length; i++){
    try {
        dataOut.write(buff[i]);
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}
dataOut.flush();
int dadosEntrada;
StringBuffer messageData = new StringBuffer();
messageData.append("\n");
while (((dadosEntrada = dataInput.read()) != -1) && !done){
    messageData.append((char) dadosEntrada);
    getMessage((char) dadosEntrada);
}

```

Figura 35: Trecho de código do envio da mensagem de requisição e recebimento dos dados.

O método de envio dos dados para o servidor está na linha `dataOut.write(buff[i]);`, sendo `buff` o buffer que armazena a mensagem de requisição e `dataOut` o canal de escrita aberto apartir da conexão estabelecida com o servidor. Já o método responsável pelo recebimento do fluxo de dados está expresso na linha `dataInput.read()`, sendo `dataInput` um canal de leitura aberto apartir da conexão estabelecida com o servidor. A forma de envio e recebimento é a mesma tanto para a solicitação da Sourcetable quanto para solicitação do recebimento dos dados para a gravação dos arquivos.

Cada dado enviado como resposta ao aplicativo, é tratado em outro método (`getMessage()`). Esse método é responsável por identificar as mensagens RTCM através dos dados que estão chegando do servidor. Após a identificação da mensagem, é feito um processo de decodificação da mesma, ou seja, é feita uma conversão dos dados binários para caracteres ASCII. Toda informação obtida por esse processo é armazenada em uma estrutura que posteriormente será escrita em um arquivo. A Figura 36 mostra o trecho de código no qual é feita a conversão. Existe um método de conversão específico para cada tipo de dado que se deseja obter.

A conversão consiste em ler a mensagem e, através da extração de *bits*, obter os valores encapsulados por ela. A quantidade de *bits* que devem ser extraídos para a obtenção de cada característica da mensagem são definidos de acordo com o que o padrão RTCM

```

private static final void loadBITS(int a){
    while(a > numbits)
    {
        if((size--)==0) break; //if(!size--)
        bitfield = (bitfield<<8)|(data[m++]);
        numbits += 8;
    }
}

private static final long getBITS(long b,int a) {
    loadBITS(a);
    b = (long)((bitfield<<(64-numbits))>>>(64-(a))); // deslocamento sem levar em conta
    bit mais significativo
    numbits -= a;
    if(b < 0){ b = ~b+1;} //realizando o complemento de 2
    return(b);
}

```

Figura 36: Trecho de código que realiza a conversão dos dados.

específica (para maiores detalhes, ver Apêndice B). Alguns cálculos podem ser realizados com os valores obtidos pela conversão, a fim de obter as unidades de medida correspondentes.

Conforme vão sendo obtidos, os valores vão sendo armazenados nos atributos de uma classe chamada RTCM3ParserData. A Figura 37 mostra parte da classe RTCM3ParserData definida na implementação. De acordo com os parâmetros passados pelo usuário, diferentes tipos de arquivos podem ser gerados. Dessa forma, o arquivo a ser aberto é definido de acordo com o que o usuário escolheu, e assim, todas as informações contidas na classe RTCM3ParserData são gravadas(escritas) no arquivo.

```

public class RTCM3ParserData {

    public char[] Message = new char[2048]; /* input-buffer */
    public int MessageSize; /* current buffer size */
    public int NeedBytes; /* bytes wanted for next run */
    public int SkipBytes; /* bytes to skip in next round */
    public long GPSWeek;
    public long GPSTOW; /* in seconds */
    public gnssdata Data = new gnssdata();
    public gpsephemeris ephemerisGPS = new gpsephemeris();
    public glonassephemeris ephemerisGLONASS = new glonassephemeris();
    public gnssdata DataNew = new gnssdata();
    public int size;
    public int[] lastlock1 = new int[64];
    public int[] lastlock2 = new int[64];
    public static boolean NO_RTCM3_MAIN = false;
    :
}

```

Figura 37: Parte inicial da definição da classe RTCM3ParserData

A Figura 38, mostra um segmento, em caracteres ASCII, de uma mensagem do tipo

1004:

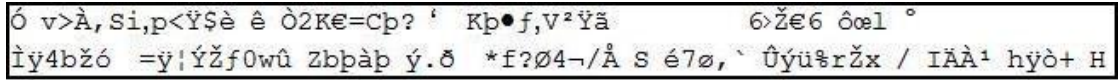


Figura 38: Mensagem do tipo 1004 do RTCM 3.

Esse segmento foi identificado, como uma mensagem, no método *getMessage()*. Para identificar esse segmento, foi utilizada a estrutura da mensagem do RTCM 3, mostrada na Figura 26 do Capítulo 2, e os caracteres representados com 8 bits. Tomando como exemplo os 3 primeiros caracteres dessa mensagem 1004, tem-se que: o primeiro deles é “Ó” sendo seu representante binário, na tabela ASCII, o valor 11010011, e que indica o prâmbulo da mensagem; em seguida tem-se um caracter não imprimível “ ” de valor binário 00000000; e por fim, o terceiro caracter “v” sendo sua representação binária 01110110. Após a leitura do terceiro caracter, é possível obter-se o tamanho da mensagem. Como na estrutura da mensagem do RTCM 3 o campo dos *bits* reservados deve possuir 6 *bits*, os dois ultimos *bits* do segundo caracter são “emprestados” aos *bits* do terceiro caracter, constituindo assim, o campo de tamanho da mensagem que deve contar 10 *bits*. Dessa maneira, após a “junção” dos *bits* desses dois caracteres, é feito o cálculo do tamanho da mensagem. Nesse exemplo, o tamanho da mensagem é 118 bytes, que é o valor decimal dos dois ultimos *bits* do segundo caracter juntos aos 8 *bits* do terceiro caracter, constituindo o valor binário 0001110110. Todos os caracteres, do quarto caracter até o antepenúltimo (os 3 caracteres finais são destinados ao CRC), são decodificados seguindo o padrão definido pelo RTCM 3 para os diferentes tipos de mensagens. Para obter o número de identificação dessa mensagem, são extraídos os 12 *bits* iniciais do campo da mensagem (do quarto caracter ao antepenúltimo), obtendo-se o valor 1004. Após essa identificação, outras características da mensagem são obtidas realizando esse mesmo procedimento de extração de *bits*, sequenciais, da mensagem.

Como especificado no padrão RINEX, para gerar os arquivos, todas as *strings* a serem gravadas são minuciosamente formatadas. A Figura 39 mostra o trecho do código com a formatação da *string* para a gravação de um dos arquivos. Essa formatação faz com que o arquivo gerado saia exatamente no formato RINEX especificado. O método *Defines.formata()* foi desenvolvido para formatar os valores especificados pelos parâmetros, toda *string* que contém valores a serem formatados são compostas por chamadas a este método. Depois da formatação, as *strings* são armazenadas em um *StringBuffer*, até serem escritas no arquivo, que encontra-se na linha *dops.write(data);*, sendo *dops* um canal de escrita aberto no arquivo.

```
temp = " "+Defines.formata("19,12",String.valueOf(handle.ephemerisGLONASS.z_pos),"double")+
Defines.formata("19,12",String.valueOf(handle.ephemerisGLONASS.z_velocity),"double")+
Defines.formata("19,12",String.valueOf(handle.ephemerisGLONASS.z_acceleration),"double")+
Defines.formata("19,12",String.valueOf(k),"double")+"\n";
temp = Defines.replace(temp, 'E', 'D');
str.append(temp);
temp = str.toString();
data = temp.getBytes();
dops.write(data);
```

Figura 39: Trecho de código que realiza a formatação de strings e grava no arquivo.

3.3 Testes

Para realizar os testes do aplicativo, o emulador escolhido foi o ClamshellCldcPhone1, que é o modelo padrão do emulador, como demonstrado na Figura 40.



Figura 40: Emulador.

Já com o aplicativo rodando, é possível o usuário visualizar os *mountpoints*, de onde ele deseja receber os dados, através da solicitação da *Sourcetable*. Após ele informar o servidor e a porta que ele deseja se conectar, o dispositivo móvel é capaz de conectar-se

ao servidor e mostrar em tela os *mountpoints* disponíveis e que enviam RTCM 3. Esta aplicação trabalha somente com o padrão RTCM versão 3, sendo este o motivo pelo qual somente os *mountpoints* que enviam RTCM 3 são mostrados em tela.



Figura 41: Resultado: recebimento da Sourcetable.

Observe a Figura 41, no primeiro emulador da figura, os parâmetros utilizados foram: servidor 200.145.185.200 e porta 2101, o que seta a *string url* de conexão como: “socket://-200.145.185.200:2101”. Após clicar em Menu, aparecerá a opção para efetuar a conexão e, assim, o dispositivo efetua a conexão (segundo emulador) e mostra na tela os *mountpoints* referentes ao servidor (terceiro emulador).

Após a visualização dos *mountpoints*, o usuário pode conectar-se a qualquer um deles, clicando em Gerar Arquivo RINEX. Uma tela de parametros é mostrada para que o usuário possa informar o servidor, a porta, o *mountpoint* ao qual deseja se conectar, o tipo de RINEX e que tipo de Arquivo ele quer obter. A Figura 42 detalha os procedimentos a serem efetuados.

Observe que, no primeiro emulador da Figura 42, o usuário deve escolher a opção “Gerar arquivo RINEX”. É possível observar, no segundo emulador da figura, que é permitido ao usuário escolher entre o RINEX versão 3 ou versão 2.1. Observa-se também



Figura 42: Procedimentos para a geração do Arquivo RINEX.

os três tipos de Arquivos: de Observação, Efeméride GPS ou Efeméride Glonass.

Ao final desses procedimentos um arquivo RINEX é gerado (terceiro emulador). Todos os arquivos gerados pelo aplicativo J2ME nos resultados dos testes estão contidos no Apendice A.

Para validar os arquivos obtidos, a aplicação original existente em C (rtcm3torinex) e o aplicativo desenvolvido neste projeto, foram executados simultaneamente. Os arquivos resultantes, de ambos os programas, foram analisados utilizando-se do aplicativo “RT_PPP”¹ (*Real Time PPP*), que está em desenvolvimento na FCT/UNESP e é capaz de realizar o processamento de arquivos RINEX para estimar as coordenadas de uma determinada estação no modo Posicionamento Por Ponto Preciso (PPP). Após a análise dos arquivos, foi feita uma comparação entre os resultados obtidos pelo processamento dos mesmos, e o resultado dessa comparação pode ser visualizado na seguinte Tabela 1:

Tabela 1: Comparação entre as precisões das coordenadas resultantes

Solução	X(m)	SIGX(m)	Y(m)	SIGY(m)	Z(m)	SIGZ(m)
AplicativoJ2ME	3687626	0,0741	-4620819	0,0844	-2386882	0,0543
RTCM3TORINEX	3687626	0,0741	-4620819	0,0844	-2386882	0,0543
Diferenças (m)	0,0001	0	0,0002	0	0,0001	0

Na tabela, é possível observar as coordenadas da estação (X,Y,Z) e suas respectivas precisões, em metros. A segunda linha, mostra as precisões resultantes da aplicação J2ME, enquanto que, na terceira linha, estão as precisões resultantes da aplicação rtcm3torinex. A comparação entre esses valores pode ser visualizada na última linha, identificada por “Diferenças”. Nessa linha, estão os resultados obtidos pela diferença entre os valores das duas aplicações. Dessa forma, analisando a tabela, é possível averiguar que existe uma pequena diferença entre os valores dos arquivos gerados. Vê-se que a diferença está abaixo do milímetro. Essa pequena diferença deve-se à maneira com a qual os valores em ponto flutuante são manipulados na plataforma. Como foi necessária a implementação de um método para a formatação de todas as strings a serem gravadas no arquivo, possivelmente alguns valores, contidos nessas strings, sofreram um arredondamento na formatação.

¹MARQUES, H. A.; MONICO, J. F. G.; SHIMABUKURO, M. H. AQUINO, M. Real Time Precise Point Positioning: Preliminary Results for the Brazilian Region. In: Proceedings of EGU General Assembly, 2010. Viena, Austria, 2010.

Capítulo 4

Conclusões

No presente trabalho, foi proposto um aplicativo, desenvolvido na plataforma J2ME, que fosse capaz de coletar dados RTCM 3, fazer a decodificação desses dados e gerar arquivos RINEX.

Com a existência de aplicações desenvolvidas na plataforma C, para *Desktop*, que são capazes de capturar dados RTCM 3 e gerar arquivos RINEX, foi sugerida então, a possibilidade do desenvolvimento de uma aplicação, com essas mesmas funcionalidades, para aparelhos móveis como celulares. A mobilidade desses aparelhos traz, aos usuários desses sistemas, grande praticidade, pois permite que ele obtenha, de onde quer que esteja, as mesmas funcionalidades das aplicações de seu *Desktop*, sendo esse o grande diferencial entre esses sistemas.

Para isso, foram realizados estudos detalhados sobre a disseminação de dados GNSS pela internet, juntamente os protocolos da rede e o protocolo NTRIP. Também foram analisados os formatos RTCM e RINEX, necessários para a implementação do aplicativo.

O aplicativo desenvolvido neste projeto validou seu objetivo principal: baseando-se em aplicações já existentes em C, transpor a funcionalidade dessas aplicações para um ambiente multiplataforma. A funcionalidade dessas aplicações consiste em coletar dados GNSS transmitidos via internet convertendo-os em arquivo RINEX. Implementando o aplicativo para a plataforma J2ME – para um dispositivo móvel, comprovou-se a possibilidade da comunicação entre o dispositivo e o servidor dos dados GNSS, e também a geração dos arquivos RINEX pelo aparelho móvel.

A aplicação desenvolvida é capaz de conectar-se a internet e receber dados, enviados pelo NTRIP, de acordo com o ponto (*mountpoint*) de onde o usuário deseja coletá-los.

Os dados recebidos encontram-se em mensagens no formato RTCM 3 e são convertidos e escritos no formato RINEX. Os arquivos RINEX obtidos pelo aplicativo, foram testados para validação de seu conteúdo e apresentam-se em conformidade com o padrão RINEX, demonstrando somente algumas pequenas variações de valores. Por essa razão, pode-se concluir que este projeto obteve resultado satisfatório, comprovando a possibilidade de ter-se em um dispositivo móvel as mesmas funcionalidades das aplicações desenvolvidas para Desktop. A partir do resultado obtido nesse projeto, é possível considerar o desenvolvimento de novas pesquisas e testes envolvendo aplicações de posicionamento por satélites em aparelhos móveis, assim como a otimização das aplicações já existentes. A seguir, são apresentadas algumas recomendações para os trabalhos a serem desenvolvidos futuramente.

4.1 Trabalhos Futuros

Durante o desenvolvimento desse projeto, surgiram vários aspectos importantes que devem/podem ser tratados no futuro:

- Otimizar a manipulação dos arquivos. Para que fosse possível a cada solicitação, escrever no arquivo sem sobrescrever o que já havia no mesmo, foi necessário que o arquivo permanecesse aberto durante toda a execução do aplicativo;
- Otimizar a formatação de *strings* utilizadas para a impressão nos arquivos, pois a plataforma, por ser bastante limitada em seus recursos, dificultou algumas tarefas realizadas, tornando-as mais dispendiosas. Foi preciso a implementação de uma função para formatar as *strings* que continham os valores a serem impressos da forma como descrito no padrão RINEX, e verificar a influência nas diferenças de valores gerados.
- Melhorar a interface gráfica utilizada, que dispoe-se de um *design* relativamente simples. Neste projeto, foram utilizados recursos mais básicos de interface, como componentes nativos de J2ME, e seria interessante tornar essa interface mais rica em detalhes e mais atrativa;
- Possibilidade de testes com outras aplicações de posicionamento por satélite em dispositivos móveis.

Referências

- COMER, D. E. *Redes de computadores e Internet*. [S.l.]: Bookman, 2007.
- COSTA, S. M. A. et al. Rbmc em tempo real, via ntrip, e seus benefícios nos levantamentos rtk e dgps. *Instituto Brasileiro de Geografia e Estatística*, p. 3–4, 2008. Disponível em: <http://www.ibge.gov.br/canal_artigos/a2008_1.php>.
- DALBELO, L. F. A. *DGPS em rede: Desenvolvimento e implantação via internet utilizando a rede gnss do estado de são paulo*. Dissertação (Mestrado) — Universidade Estadual Paulista, Faculdade de Ciências e Tecnologia, 2010.
- DETTMERING, D.; WAESE, C.; WEBER, G. *Ntrip, Version 1.0: Example implementation*. Frankfurt, Germany, 2006.
- GMBH, D. S. A. *Manual for rtcm3torinex*. Frankfurt, Germany, 2009. Disponível em: <http://igs.bkg.bund.de/index_ntrip.htm>.
- GMBH, G. *Geo++ GNMoblie*. 2010. Disponível em: <<http://geopp.de/>>.
- GURTNER, W.; ESTEY, L. *RINEX The Receiver Independent Exchange Format. Version 3*. [S.l.], November 2007.
- JOHNSON, T. M. *Java para Dispositivos Móveis: desenvolvendo aplicações com j2me*. [S.l.]: Novatec Editora, 2007.
- KUROSE J. F. E ROSS, K. W. *Redes de computadores e a Internet: Uma abordagem top-down*. [S.l.]: Person Addison Weseleys, 2005.
- MUCHOW, J. W. *Core J2ME: Tecnologia & midp*. [S.l.]: Pearson Makron Books, 2007.
- ROMEIRO, B. G. B. de A. *Desenvolvimento de aplicativos para dispositivos móveis na plataforma J2ME*. 2005. Disponível em: <<http://dsc.upe.br/tcc/20051.html>>.
- RTCM, S. C. N. . *Networked Transport of RTCM via Internet Protocol (NTRIP): Version 1.0*. [S.l.], 2005. Disponível em: <igs.bkg.bund.de/root_ftp/NTRIP/documentation-/NtripDocumentation.pdf>.
- RTCM, S. C. N. . *RTCM STANDARD 10403.1 FOR DIFFERENTIAL GNSS (GLOBAL NAVIGATION SATELLITE SYSTEMS) SERVICES. Version 3*. Virginia, 2006.

RTCM, S. C. N. . *RTCM*: Rtcn standard 10410.1 networked transport of rtcn via internet protocol(ntrip) - version 2.0. Frankfurt, Germany, 2009.

SUN. *Java Platform, Micro Edition Software Development Kit 3.0*. 2009. Disponível em: <<http://www.oracle.com/technetwork/java/javame/javamobile/download/sdk/index.html>>.

APÊNDICE A – Arquivos RINEX gerados pela aplicação

De acordo com os parâmetros passados pelo usuário para o aplicativo diferentes tipos de arquivos RINEX podem ser gerados. O usuário pode escolher entre: Arquivo de Observação, Arquivo de navegação GPS ou Arquivo de navegação GLONASS. Além de escolher o tipo de arquivo que desejado, o usuário pode escolher entre as versões do RINEX: 3.00 ou 2.11. A seguir, estão especificados os diferentes arquivos gerados pela aplicação, conforme os parâmetros especificados pelo usuário.

```

2.11      OBSERVATION DATA      M (Mixed)      RINEX VERSION / TYPE
RTCM3TORINEX 1.37      2011-07-29 00:27      PGM / RUN BY / DATE
RTCM3TORINEX      MARKER NAME
OBSERVER / AGENCY
REC # / TYPE / VERS
ANT # / TYPE
APPROX POSITION XYZ
ANTENNA: DELTA H/E/N
WAVELENGTH FACT L1/2
# / TYPES OF OBSERV
TIME OF FIRST OBS
END OF HEADER

      .0000      .0000      .0000
      .0000      .0000      .0000
1      1
000007      C1      C2      P2      L1      L2      S1      S2
2011      07      29      03      27      24.0000000      GPS

11 07 29 03 27 24.0000000 0 019G15G07G08G26G28G04G27G09G05G12G17R17
R23R03R14R02R12R13R24
22155102.054      22155106.934      116425879.923 9 90721458.008 9
50.250      45.750
25049295.573      25049296.114      131634929.000 8 102572677.267 5
34.750      21.500
23466725.184      23466731.004      123318572.333 9 96092436.409 7
46.500      29.750
20498050.823      20498056.323      107718000.751 9 83936107.993 9
51.250      44.500
22212796.451      22212800.812      116729086.113 9 90957743.894 7
48.500      31.500
24252538.480      24252541.180      127447963.000 9 99310086.326 3
37.500      15.500
23426909.283      23426915.564      123109233.231 9 95929273.429 5
39.000      22.000
25349595.232      133213013.812 7
31.750
24248855.460      24248858.820      127428581.205 9 99294999.027 8
36.500      34.250
25426819.272      25426829.012      133618851.61019 104118586.21419
37.750      38.000
20282130.526      20282135.806      106583319.956 9 83051946.679 9
51.750      52.500
23759948.283      23759949.123      127144246.41119 98889986.60317
37.500      30.000
20405573.043      20405578.323      109156069.236 9 84899186.353 9
54.000      48.750
22323014.271      22323020.731      119036117.666 9 92583649.887 9
45.000      43.000
22573630.611      120330166.420 9
43.750
24255415.360      24255418.060      129431503.403 9 100668942.862 9
40.500      36.500
23070162.167      23070169.327      123236739.43119 95850823.66819
44.500      40.500
21121209.340      21121211.680      112786149.087 9 87722562.301 9
51.750      45.000
19943960.608      19943964.968      106649352.740 9 82949511.176 9
54.500      50.500
11 07 29 03 27 25.0000000 0 019G15G07G08G26G28G04G27G09G05G12G17R17
R23R03R14R02R12R13R24
22154658.274      22154663.334      116423543.955 9 90719637.771 9
50.500      45.750
25049676.694      25049676.294      131636930.485 8 102574236.876 6
34.500      26.000
23466986.884      23466992.884      123319947.691 9 96093508.119 7
46.500      29.500
20497881.764      20497888.043      107717113.719 9 83935416.797 9

```

Figura A.1: Arquivo de dados de Observação RINEX 2.11

```

      3.00      OBSERVATION DATA      M (Mixed)      RINEX VERSION / TYPE
RTCM3TORINEX 1.37      20110729 020009 UTC      PGM / RUN BY / DATE
RTCM3TORINEX      MARKER NAME
GEODETIC      MARKER TYPE
      OBSERVER / AGENCY
      REC # / TYPE / VERS
      ANT # / TYPE
      APPROX POSITION XYZ
      ANTENNA: DELTA H/E/N
      SYS / # / OBS TYPES
      SYS / # / OBS TYPES
      SYS / # / OBS TYPES
      TIME OF FIRST OBS
      END OF HEADER

      .0000      .0000      .0000
      .0000      .0000      .0000
G 009 C1C L1C S1C C2P L2P S2P C2X L2X S2X
R 009 C1C L1C S1C C2P L2P S2P C2C L2C S2C
S 003 C1C L1C S1C
      2011      07      29      05      00      24.0000000      GPS

> 2011 07 29 05 00 24.0000000 0 017
G15 21384963.878 112378773.400 9 49.500
21384969.757 87567868.121 9 48.250
G08 25532996.410 134176805.279 9 36.500
G26 21461424.917 112780561.848 9 50.500 21461431.798
87880959.489 9 37.000
G02 22741620.409 119508034.492 9 44.000 22741622.990
93123146.237 7 28.750
G28 24786981.656 130256463.220 7 31.000
G04 22411710.211 117774334.427 9 47.000 22411717.251
91772210.216 9 36.000
G27 20486268.323 107656067.094 9 51.000 20486274.143
83887847.063 9 42.000
G09 22642071.909 118984893.086 9 49.500 22642078.589
92715503.880 8 35.750
G12 23616577.264 124105972.335 9 43.750
23616583.484 9 6705958.157 9 41.000
G17 22060728.774 115929913.980 9 47.250
22060733.594 9 0335006.874 9 46.250
R17 19978349.90819 106908179.54419 50.50019 19978353.20819
83150828.19419 44.00019
R23 23756394.424 9 127080696.208 9 45.000 9 23756399.484 9
98840562.600 9 38.250 9
R03 20245236.908 9 107956492.265 9 54.250 9 20245241.488 9
83966164.381 9 48.000 9
R14 20325547.028 0 108346612.974 0 51.500 0
R18 24067228.080 0 128472536.978 0 26.250 0
R13 21204692.660 9 113231948.484 9 50.000 9 21204696.299 9
88069294.937 9 44.750 9
R24 19719494.092 9 105449029.776 9 55.250 9 19719498.092 9
82015926.808 9 50.250 9
> 2011 07 29 05 00 25.0000000 0 016
G15 21385149.698 112379751.869 9 50.000
21385155.917 87568630.576 9 48.250
G08 25533481.650 134179359.195 9 36.500
G26 21461901.237 112783064.528 9 50.500 21461908.218
87882909.621 9 37.000
G02 22741045.249 119505009.242 9 43.500 22741047.549
93120788.897 7 29.500
G28 24787405.455 130258691.770 7 31.000
G04 22411501.751 117773237.254 9 47.500 22411508.031
91771355.273 8 35.500
G27 20485859.323 107653918.857 9 51.000 20485865.823
83886173.111 9 41.750
G09 22641563.909 118982223.150 9 48.750 22641570.310
92713423.404 8 35.500

```

Figura A.2: Arquivo de dados de Observação RINEX 3.0

```

1      2.10      N: GPS NAV DATA      RINEX VERSION / TYPE
RTCM3TORINEX 1.37      2011-07-29 01:38      PGM / RUN BY / DATE
END OF HEADER
02 11 07 29 06 00 0.0 3.488301299512D-4 1.818989403545D-12 0.000000000000D+0
1.030000000000D+2 2.168750000000D+1 5.642735042563D-9 -0.237936738174D+0
1.244246959686D-6 0.010371618322D+0 5.424022674560D-6 -3.038220096588D+3
4.536000000000D+5 1.136213541030D-7 -2.014963414925D+0 3.166496753692D-8
0.938638241181D+0 2.629062500000D+2 -2.956963822212D+0 -8.592143611555D-9
4.321608583772D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -1.722946763038D-8 1.030000000000D+2
0.000000000000D+0 0.000000000000D+0
04 11 07 29 06 00 0.0 1.857709139585D-4 1.091393642127D-11 0.000000000000D+0
7.200000000000D+1 1.971875000000D+1 5.533801933634D-9 2.992812647299D+0
9.201467037200D-7 0.009630798012D+0 5.675479769706D-6 -3.038220062057D+3
4.536000000000D+5 -6.332993507385D-8 -1.997673657206D+0 -1.527369022369D-7
0.937712123336D+0 2.597187500000D+2 0.740278753295D+0 -8.254629552739D-9
3.921591921473D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -6.519258022308D-9 7.200000000000D+1
0.000000000000D+0 0.000000000000D+0
05 11 07 29 06 00 0.0 -1.961290836334D-4 -5.456968210637D-12 0.000000000000D+0
8.000000000000D+1 -2.750000000000D+1 5.016994692252D-9 1.805586707076D+0
-1.197680830955D-6 0.002272210782D+0 8.454546332359D-6 -3.038285335540D+3
4.536000000000D+5 0.000000000000D+0 -0.952471667685D+0 6.519258022308D-8
0.953343296220D+0 2.136562500000D+2 0.159367074209D+0 -8.149268021152D-9
-6.510985494394D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -1.024454832077D-8 8.000000000000D+1
0.000000000000D+0 0.000000000000D+0
07 11 07 29 04 00 0.0 1.976499333977D-5 1.591615728102D-12 0.000000000000D+0
9.000000000000D+0 6.921875000000D+1 4.339823628216D-9 -2.765835964101D+0
3.587454557418D-6 0.004764974582D+0 1.188367605209D-6 -3.038354660034D+3
4.464000000000D+5 -9.126961231231D-8 1.153346685633D+0 2.048909664154D-8
0.974691397889D+0 3.653125000000D+2 -3.119240207457D+0 -8.273201754918D-9
2.064371703653D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -1.071020960807D-8 9.000000000000D+0
0.000000000000D+0 0.000000000000D+0
08 11 07 29 06 00 0.0 8.639879524707D-6 1.136868377216D-13 0.000000000000D+0
8.000000000000D+1 4.806250000000D+1 3.923734867878D-9 -2.213965179439D+0
2.415850758552D-6 0.011826995178D+0 1.052394509315D-6 -3.038424325942D+3
4.536000000000D+5 -1.341104507446D-7 1.230151293584D+0 -1.396983861923D-7
0.996429232637D+0 3.801875000000D+2 -3.063906164972D+0 -8.062478691742D-9
2.028655930233D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -4.190951585769D-9 8.000000000000D+1
0.000000000000D+0 0.000000000000D+0
09 11 07 29 06 00 0.0 9.575160220265D-5 2.287423592153D-12 0.000000000000D+0
5.400000000000D+1 6.946875000000D+1 3.980522947615D-9 -2.439708697305D+0
3.729015588760D-6 0.017409036401D+0 1.030042767524D-6 -3.038333911895D+3
4.536000000000D+5 2.086162567138D-7 1.125298685412D+0 7.823109626770D-8
0.983592297652D+0 3.707500000000D+2 1.590157314652D+0 -7.853184259503D-9
1.646497154643D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -5.587935447692D-9 5.400000000000D+1
0.000000000000D+0 0.000000000000D+0
10 11 07 29 04 00 0.0 -8.253380656242D-6 -1.136868377216D-12 0.000000000000D+0
8.700000000000D+1 -2.325000000000D+1 5.093426447370D-9 0.967999972558D+0
-8.996576070785D-7 0.010253466200D+0 8.609145879745D-6 -3.038380008697D+3
4.464000000000D+5 -8.940696716308D-8 -0.929080230800D+0 9.685754776000D-8
0.948300206521D+0 2.079062500000D+2 0.705175916950D+0 -8.304274477793D-9
-7.975332204598D-10 1.646000000000D+3 0.000000000000D+0
0.000000000000D+0 -2.328306436538D-9 8.700000000000D+1
0.000000000000D+0 0.000000000000D+0
12 11 07 29 06 00 0.0 1.227390021085D-5 2.501110429875D-12 0.000000000000D+0
7.000000000000D+0 -1.276562500000D+2 3.970522531057D-9 -1.923260785725D+0

```

Figura A.3: Arquivo de Mensagem NAVEGAÇÃO GPS RINEX 2.10

```

2.10          G: GLONASS NAV DATA          RINEX VERSION / TYPE
RTCM3TORINEX 1.37          2011-07-29 01:48  PGM / RUN BY / DATE
END OF HEADER
02 11 07 29 03 45 0.0 -2.447171136736D-4 -1.818989403545D-12 1.314000000000D+4
6.600254882812D+3 0.677042007446D+0 0.000000000000D+0 0.000000000000D+0
-1.354001806640D+4 2.734848022460D+0 0.000000000000D+0 -4.000000000000D+0
-1.217483789062D+4 1.582432700561D+0 -2.793967723846D-9 0.000000000000D+0
03 11 07 29 04 45 0.0 4.824250936508D-5 2.728484105318D-12 1.725000000000D+4
1.115419189453D+4 2.854516029357D+0 0.000000000000D+0 0.000000000000D+0
-9.571187988281D+3 -0.149222373962D+0 -4.656612873077D-9 -6.000000000000D+0
1.187088867187D+4 1.585911750793D+0 0.000000000000D+0 0.000000000000D+0
12 11 07 29 04 45 0.0 -7.912516593933D-6 1.818989403545D-12 1.719000000000D+4
-9.263110351562D+3 -1.242962837219D+0 -2.793967723846D-9 0.000000000000D+0
4.414837890625D+3 -0.192471504211D+0 9.313225746154D-10 -1.000000000000D+0
9.078952148437D+3 3.282320022583D+0 0.000000000000D+0 0.000000000000D+0
13 11 07 29 04 45 0.0 -3.141993656754D-4 -9.094947017729D-13 1.725000000000D+4
-8.999421386718D+3 1.138129234313D+0 -1.862645149230D-9 0.000000000000D+0
-4.098337402343D+3 -0.213360786437D+0 -1.862645149230D-9 -2.000000000000D+0
-8.352081054687D+3 3.351419448852D+0 0.000000000000D+0 0.000000000000D+0
14 11 07 29 04 45 0.0 -1.882351934909D-4 -9.094947017729D-13 1.725000000000D+4
9.909357421875D+3 2.919425964355D+0 0.000000000000D+0 0.000000000000D+0
-9.883980468750D+3 -0.114113807678D+0 -4.656612873077D-9 -7.000000000000D+0
1.145145361328D+4 1.418858528137D+0 0.000000000000D+0 0.000000000000D+0
17 11 07 29 04 45 0.0 -2.634786069393D-4 1.818989403545D-12 1.725000000000D+4
1.168197802734D+4 0.038569450378D+0 0.000000000000D+0 0.000000000000D+0
1.022759716796D+4 -0.436289787292D+0 -3.725290298461D-9 4.000000000000D+0
2.900080566406D+3 -3.545433044433D+0 -2.793967723846D-9 0.000000000000D+0
23 11 07 29 04 45 0.0 -3.042481839656D-4 -3.637978807091D-12 1.725000000000D+4
1.084971630859D+4 -1.567658424377D+0 -9.313225746154D-10 0.000000000000D+0
2.748402343750D+3 2.718855857849D+0 -2.793967723846D-9 3.000000000000D+0
9.841814941406D+3 -0.416167259216D+0 1.862645149230D-9 0.000000000000D+0
24 11 07 29 04 45 0.0 8.875317871570D-5 9.094947017729D-13 1.725000000000D+4
1.573948583984D+4 -1.100672721862D+0 -9.313225746154D-10 0.000000000000D+0
-1.388061376953D+4 1.657581329345D+0 -4.656612873077D-9 2.000000000000D+0
-1.452036425781D+4 -2.781159400939D+0 0.000000000000D+0 0.000000000000D+0

```

Figura A.4: Arquivo de Mensagem NAVEGAÇÃO GLONASS RINEX 2.10

```

1      3.00          N: GNSS NAV DATA      M: Mixed          RINEX VERSION / TYPE
RTCM3TORINEX 1.37          20110805 181548 UTC PGM / RUN BY / DATE
                                END OF HEADER
G03 2011 08 05 20 00 00 7.511372677981D-4 5.002220859751D-12 0.000000000000D+0
      2.800000000000D+1 -3.268750000000D+1 5.796312868268D-9 1.249681788296D+0
      -1.754611730575D-6 0.014606265584D+0 6.683170795440D-6 -3.038329666366D+3
      5.040000000000D+5 -3.290014171600D-7 2.983651293989D+0 2.980232238769D-8
      0.929554135798D+0 2.400937500000D+2 1.115386322625D+0 -8.802509516996D-9
      -8.786080261223D-11 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -4.656612873077D-9 2.800000000000D+1
      0.000000000000D+0 0.000000000000D+0
G04 2011 08 05 22 00 00 1.930175349116D-4 1.091393642127D-11 0.000000000000D+0
      5.300000000000D+1 1.134375000000D+1 5.064853828634D-9 -0.924376265993D+0
      5.811452865600D-7 0.009658697177D+0 1.059472560882D-5 -3.038316797256D+3
      5.112000000000D+5 1.639127731323D-7 -2.123571425970D+0 1.154839992523D-7
      0.937931802430D+0 1.622500000000D+2 0.742997646525D+0 -8.238914612435D-9
      -4.318037006430D-10 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -6.519258022308D-9 5.300000000000D+1
      0.000000000000D+0 0.000000000000D+0
G07 2011 08 05 22 00 00 2.080807462334D-5 1.591615728102D-12 0.000000000000D+0
      1.100000000000D+1 -7.196875000000D+1 4.031953661339D-9 0.651275455185D+0
      -3.669410943984D-6 0.004810767131D+0 8.586794137954D-6 -3.038350482940D+3
      5.112000000000D+5 1.005828380584D-7 1.027558885883D+0 8.940696716308D-8
      0.974879600837D+0 2.213750000000D+2 -3.116086210206D+0 -7.944973797191D-9
      3.464430021702D-11 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -1.071020960807D-8 1.100000000000D+1
      0.000000000000D+0 0.000000000000D+0
G08 2011 08 05 21 59 44 8.739065378904D-6 1.136868377216D-13 0.000000000000D+0
      1.600000000000D+1 -6.821875000000D+1 3.589435228670D-9 0.154996084186D+0
      -3.458932042121D-6 0.011874510557D+0 8.946284651756D-6 -3.038419569015D+3
      5.111840000000D+5 1.918524503707D-7 1.104563427923D+0 4.284083843231D-8
      0.996631817532D+0 2.256562500000D+2 -3.061554070655D+0 -7.836755003730D-9
      9.464679956196D-11 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -4.190951585769D-9 1.600000000000D+1
      0.000000000000D+0 0.000000000000D+0
G10 2011 08 05 22 00 00 -9.056180715560D-6 -1.136868377216D-12 0.000000000000D+0
      4.000000000000D+0 1.030937500000D+2 4.662327062195D-9 -1.8907673224174D+0
      5.306676030158D-6 0.010280598071D+0 7.443130016326D-6 -3.038371934890D+3
      5.112000000000D+5 6.705522537231D-8 -1.055166778903D+0 -1.788139343261D-7
      0.948188405933D+0 2.299375000000D+2 0.703239040437D+0 -8.084979628996D-9
      1.117903708033D-10 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -2.328306436538D-9 4.000000000000D+0
      0.000000000000D+0 0.000000000000D+0
G11 2011 08 05 22 00 00 -1.743733882904D-4 -3.296918293926D-12 0.000000000000D+0
      6.400000000000D+1 -8.937500000000D+0 6.474912563240D-9 0.803475886830D+0
      -2.142041921615D-7 0.012030480895D+0 9.473413228988D-6 -3.03832778930D+3
      5.112000000000D+5 -4.842877388000D-8 -2.381768713495D+0 1.378357410430D-7
      0.888047267304D+0 1.595000000000D+2 0.949971652332D+0 -9.076092341390D-9
      -4.778770483543D-10 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -1.164153218269D-8 6.400000000000D+1
      0.000000000000D+0 0.000000000000D+0
G13 2011 08 05 22 00 00 2.577533014118D-4 -1.818989403545D-12 0.000000000000D+0
      4.900000000000D+1 -1.031250000000D+1 3.924092025612D-9 3.105390725428D+0
      -4.600733518600D-7 0.004696983378D+0 4.056841135025D-6 -3.038434320449D+3
      5.112000000000D+5 -2.980232238769D-8 0.053897108756D+0 1.117587089538D-8
      0.989259762251D+0 3.164375000000D+2 1.911717621843D+0 -7.956760002420D-9
      -1.914365455291D-10 1.647000000000D+3 0.000000000000D+0
      0.000000000000D+0 -1.117587089538D-8 4.900000000000D+1
      0.000000000000D+0 0.000000000000D+0
G16 2011 08 05 22 00 00 -1.899520866572D-4 -2.387423592153D-12 0.000000000000D+0
      2.600000000000D+1 -6.468750000000D+0 4.327680265253D-9 -1.927242531284D+0

```

Figura A.5: Arquivo de Mensagem de Navegação GNSS Rinex 3.0 (Mixed GPS / GLONASS). Este arquivo contém dados sobre dois sistemas de satélites: o GPS e o GLONASS, comprovando que os arquivos RINEX 3 podem conter dados de mais de um sistema de satélite.

APÊNDICE B – Tipos de mensagens RTCM 3 abordadas nesse projeto

Para a decodificação das mensagens RTCM, foram estudados os diferentes tipos de estruturas que cada uma das mensagens possui. A seguir, estão listadas as estruturas das mensagens que foram decodificadas nesse projeto.

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
Message Number (e.g., “1001”= 0011 1110 1001)	DF002	uint12	12
Reference Station ID	DF003	uint12	12
GPS Epoch Time (TOW)	DF004	uint30	30
Synchronous GNSS Flag	DF005	bit(1)	1
No. of GPS Satellite Signals Processed	DF006	uint5	5
GPS Divergence-free Smoothing Indicator	DF007	bit(1)	1
GPS Smoothing Interval	DF008	bit(3)	3
<i>TOTAL</i>			<i>64</i>

Figura B.1: Cabeçalho das mensagens 1001-1004. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GPS Satellite ID	DF009	uint6	6
GPS L1 Code Indicator	DF010	bit(1)	1
GPS L1 Pseudorange	DF011	uint24	24
GPS L1 PhaseRange – L1 Pseudorange	DF012	int20	20
GPS L1 Lock time Indicator	DF013	uint7	7
TOTAL			58

Figura B.2: Conteúdo da mensagem 1001. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GPS Satellite ID	DF009	uint6	6
GPS L1 Code Indicator	DF010	bit(1)	1
GPS L1 Pseudorange	DF011	uint24	24
GPS L1 PhaseRange – L1 Pseudorange	DF012	int20	20
GPS L1 Lock time Indicator	DF013	uint7	7
GPS Integer L1 Pseudorange Modulus Ambiguity	DF014	uint8	8
GPS L1 CNR	DF015	uint8	8
TOTAL			74

Figura B.3: Conteúdo da mensagem 1002. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GPS Satellite ID	DF009	uint6	6
GPS L1 Code Indicator	DF010	bit(1)	1
GPS L1 Pseudorange	DF011	uint24	24
GPS L1 PhaseRange – L1 Pseudorange	DF012	int20	20
GPS L1 Lock time Indicator	DF013	uint7	7
GPS L2 Code Indicator	DF016	bit(2)	2
GPS L2-L1 Pseudorange Difference	DF017	int14	14
GPS L2 PhaseRange – L1 Pseudorange	DF018	int20	20
GPS L2 Lock time Indicator	DF019	uint7	7
TOTAL			101

Figura B.4: Conteúdo da mensagem 1003. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GPS Satellite ID	DF009	uint6	6
GPS L1 Code Indicator	DF010	bit(1)	1
GPS L1 Pseudorange	DF011	uint24	24
GPS L1 PhaseRange – L1 Pseudorange	DF012	int20	20
GPS L1 Lock time Indicator	DF013	uint7	7
GPS Integer L1 Pseudorange Modulus Ambiguity	DF014	uint8	8
GPS L1 CNR	DF015	uint8	8
GPS L2 Code Indicator	DF016	bit(2)	2
GPS L2-L1 Pseudorange Difference	DF017	int14	14
GPS L2 PhaseRange – L1 Pseudorange	DF018	int20	20
GPS L2 Lock time Indicator	DF019	uint7	7
GPS L2 CNR	DF020	uint8	8
TOTAL			125

Figura B.5: Conteúdo da mensagem 1004. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
Message Number (“1009”=0011 1111 0001)	DF002	uint12	12
Reference Station ID	DF003	uint12	12
GLONASS Epoch Time (t_k)	DF034	uint27	27
Synchronous GNSS Flag	DF005	bit(1)	1
No. of GLONASS Satellite Signals Processed	DF035	uint5	5
GLONASS Divergence-free Smoothing Indicator	DF036	bit(1)	1
GLONASS Smoothing Interval	DF037	bit(3)	3
TOTAL			61

Figura B.6: Cabeçalho das mensagens RTCM 1009-1012. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GLONASS Satellite ID (Satellite Slot Number)	DF038	uint6	6
GLONASS Code Indicator	DF039	bit(1)	1
GLONASS Satellite Frequency Channel Number	DF040	uint5	5
GLONASS L1 Pseudorange	DF041	uint25	25
GLONASS L1 PhaseRange – L1 Pseudorange	DF042	int20	20
GLONASS L1 Lock time Indicator	DF043	uint7	7
TOTAL			64

Figura B.7: Conteúdo da mensagem 1009. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GLONASS Satellite ID (Satellite Slot Number)	DF038	uint6	6
GLONASS L1 Code Indicator	DF039	bit(1)	1
GLONASS Satellite Frequency Channel Number	DF040	uint5	5
GLONASS L1 Pseudorange	DF041	uint25	25
GLONASS L1 PhaseRange – L1 Pseudorange	DF042	int20	20
GLONASS L1 Lock time Indicator	DF043	uint7	7
GLONASS Integer L1 Pseudorange Modulus Ambiguity	DF044	uint7	7
GLONASS L1 CNR	DF045	uint8	8
TOTAL			79

Figura B.8: Conteúdo da mensagem 1010. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GLONASS Satellite ID (Satellite Slot Number)	DF038	uint6	6
GLONASS L1 Code Indicator	DF039	bit(1)	1
GLONASS Satellite Frequency Channel Number	DF040	uint5	5
GLONASS L1 Pseudorange	DF041	uint25	25
GLONASS L1 PhaseRange – L1 Pseudorange	DF042	int20	20
GLONASS L1 Lock time Indicator	DF043	uint7	7
GLONASS L2 Code Indicator	DF046	bit(2)	2
GLONASS L2-L1 Pseudorange Difference	DF047	uint14	14
GLONASS L2 PhaseRange – L1 Pseudorange	DF048	int20	20
GLONASS L2 Lock time Indicator	DF049	uint7	7
TOTAL			107

Figura B.9: Conteúdo da mensagem 1011. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS
GLONASS Satellite ID (Satellite Slot Number)	DF038	uint6	6
GLONASS L1 Code Indicator	DF039	bit(1)	1
GLONASS Satellite Frequency Channel Number	DF040	uint5	5
GLONASS L1 Pseudorange	DF041	uint25	25
GLONASS L1 PhaseRange – L1 Pseudorange	DF042	int20	20
GLONASS L1 Lock time Indicator	DF043	uint7	7
GLONASS Integer L1 Pseudorange Modulus Ambiguity	DF044	uint7	7
GLONASS L1 CNR	DF045	uint8	8
GLONASS L2 Code Indicator	DF046	bit(2)	2
GLONASS L2-L1 Pseudorange Difference	DF047	uint14	14
GLONASS L2 PhaseRange – L1 Pseudorange	DF048	int20	20
GLONASS L2 Lock time Indicator	DF049	uint7	7
GLONASS L2 CNR	DF050	uint8	8
TOTAL			130

Figura B.10: Conteúdo da mensagem 1012. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS	NOTES
Message Number	DF002	uint12	12	1019
GPS Satellite ID	DF009	uint6	6	
GPS Week Number	DF076	uint10	10	0 - 1023
GPS SV ACCURACY	DF077	uint4	4	See GPS SPS Signal Specification, 2.4.3
GPS CODE ON L2	DF078	bit(2)	2	00 = Reserved 01 = P code on 10 = C/A code on 11 = L2C on
GPS IDOT	DF079	int14	14	See GPS SPS Signal Specification, 2.4.3
GPS IODE	DF071	uint8	8	See GPS SPS Signal Specification, 2.4.3
GPS t_{oc}	DF081	uint16	16	See GPS SPS Signal Specification, 2.4.3
GPS a_{t2}	DF082	int8	8	See GPS SPS Signal Specification, 2.4.3
GPS a_{t1}	DF083	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS a_{t0}	DF084	int22	22	See GPS SPS Signal Specification, 2.4.3
GPS IODC	DF085	uint10	10	See GPS SPS Signal Specification, 2.4.3
GPS C_{rs}	DF086	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS Δn (DELTA n)	DF087	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS M_0	DF088	int32	32	See GPS SPS Signal Specification, 2.4.3
GPS C_{uc}	DF089	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS Eccentricity (e)	DF090	uint32	32	See GPS SPS Signal Specification, 2.4.3
GPS C_{us}	DF091	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS $(A)^{1/2}$	DF092	uint32	32	See GPS SPS Signal Specification, 2.4.3
GPS t_{oe}	DF093	uint16	16	See GPS SPS Signal Specification, 2.4.3
GPS C_{ic}	DF094	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS Ω_0 (OMEGA) ₀	DF095	int32	32	See GPS SPS Signal Specification, 2.4.3
GPS C_{is}	DF096	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS i_0	DF097	int32	32	See GPS SPS Signal Specification, 2.4.3
GPS C_{rc}	DF098	int16	16	See GPS SPS Signal Specification, 2.4.3
GPS ω (Argument of Perigee)	DF099	int32	32	See GPS SPS Signal Specification, 2.4.3
GPS OMEGADOT (Rate of Right Ascension)	DF100	int24	24	See GPS SPS Signal Specification, 2.4.3
GPS t_{GD}	DF101	int8	8	See GPS SPS Signal Specification, 2.4.3
GPS SV HEALTH	DF102	uint6	6	See GPS SPS Signal Specification, 2.4.3
GPS L2 P data flag	DF103	bit(1)	1	0: L2 P-Code NAV data ON 1: L2 P-Code NAV data OFF
GPS Fit Interval	DF137	bit(1)	1	See GPS SPS Signal Specification, 2.4.3
TOTAL			488	

Figura B.11: Conteúdo da mensagem 1019. (RTCM, 2006)

DATA FIELD	DF NUMBER	DATA TYPE	NO. OF BITS	NOTES
Message Number	DF002	uint12	12	1020
GLONASS Satellite ID (Satellite Slot Number)	DF038	uint6	6	
GLONASS Satellite Frequency Channel Number	DF040	uint5	5	
GLONASS almanac health (C_n word)	DF104	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS almanac health availability indicator	DF105	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS P1	DF106	bit(2)	2	See GLONASS ICD Version 5.0
GLONASS t_k	DF107	bit(12)	12	See GLONASS ICD Version 5.0
GLONASS MSB of B_n word	DF108	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS P2	DF109	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS t_b	DF110	uint7	7	See GLONASS ICD Version 5.0
GLONASS $x_n(t_b)$, first derivative	DF111	intS24	24	See GLONASS ICD Version 5.0
GLONASS $x_n(t_b)$	DF112	intS27	27	See GLONASS ICD Version 5.0
GLONASS $x_n(t_b)$, second derivative	DF113	intS5	5	See GLONASS ICD Version 5.0
GLONASS $y_n(t_b)$, first derivative	DF114	intS24	24	See GLONASS ICD Version 5.0
GLONASS $y_n(t_b)$	DF115	intS27	27	See GLONASS ICD Version 5.0
GLONASS $y_n(t_b)$, second derivative	DF116	intS5	5	See GLONASS ICD Version 5.0
GLONASS $z_n(t_b)$, first derivative	DF117	intS24	24	See GLONASS ICD Version 5.0
GLONASS $z_n(t_b)$	DF118	intS27	27	See GLONASS ICD Version 5.0
GLONASS $z_n(t_b)$, second derivative	DF119	intS5	5	See GLONASS ICD Version 5.0
GLONASS P3	DF120	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS $\gamma_n(t_b)$	DF121	intS11	11	See GLONASS ICD Version 5.0
GLONASS-M P	DF122	bit(2)	2	See GLONASS ICD Version 5.0
GLONASS-M l_n (third string)	DF123	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS $\tau_n(t_b)$	DF124	intS22	22	See GLONASS ICD Version 5.0
GLONASS-M $\Delta\tau_n$	DF125	intS5	5	See GLONASS ICD Version 5.0
GLONASS E_n	DF126	uint5	5	See GLONASS ICD Version 5.0
GLONASS-M P4	DF127	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS-M F_T	DF128	uint4	4	See GLONASS ICD Version 5.0
GLONASS-M N_T	DF129	uint11	11	See GLONASS ICD Version 5.0
GLONASS-M M	DF130	bit(2)	2	See GLONASS ICD Version 5.0
GLONASS The Availability of Additional Data	DF131	bit(1)	1	See GLONASS ICD Version 5.0
GLONASS N^A	DF132	uint11	11	See GLONASS ICD Version 5.0
GLONASS τ_c	DF133	intS32	32	See GLONASS ICD Version 5.0
GLONASS-M N_4	DF134	uint5	5	See GLONASS ICD Version 5.0
GLONASS-M τ_{GPS}	DF135	intS22	22	See GLONASS ICD Version 5.0
GLONASS-M l_n (fifth string)	DF136	bit(1)	1	See GLONASS ICD Version 5.0
Reserved		bit(7)	7	
TOTAL			360	

Figura B.12: Conteúdo da mensagem 1020. (RTCM, 2006)