Range Networks

OpenBTS P2.8

Users' Manual

Doc. Rev. 1

Contents

1 Introduction **8**

1.1 Scope and Audience 9

1.2 Disclaimers 9

1.2.1 Warranty 9

1.2.2 Accuracy 9

1.2.3 Patent Laws 9

1.2.4 Trademarks 9

1.2.5 Telecom and Radio Spectrum Laws 9

1.2.6 Software Licensing and Distribution 10

1.3 Abbreviations 10

1.4 References 11

1.4.1 ETSI/3GPP 11

1.4.2 Unix Documentation 12

1.4.3 IETF 12

1.5 Contact Information 13

2 The OpenBTS Application Suite **14**

2.1 OpenBTS 14

2.2 Transceiver 15

2.3 Asterisk 15

2.4 Subscriber Registry 15

2.5 Smqueue 15

2.6 Network Organization 16

3 The OpenBTS GSM Air Interface **17**

3.1 Um Layers . . . 18
3.1.1 Physical Layer (L1) . . . 18
3.1.2 Data Link Layer (L2) . . . 19
3.1.3 Network Layer (L3) . . . 19
3.2 Um logical channels . . . 20
3.2.1 Traffic channels (TCH) . . . 20
3.2.2 Dedicated Control Channels (DCCHs) . . . 20
3.2.3 Non-Dedicated Control Channels (NDCCHs) . . . 21
3.2.4 Allowed channel combinations . . . 22
3.3 Service Capacity of Um . . . 22

4 External Databases **24**
4.1 Editing Sqlite3 Databases . . . 24
4.2 The Configuration Table . . . 24
4.3 TMSI Table . . . 31
4.4 Channel Table . . . 32

5 Processes Inside OpenBTS **34**
5.1 T3122 Exponential Back-Off . . . 34
5.2 Downlink Power and Congestion Management . . . 34
5.3 Uplink Power and Timing Control . . . 35
5.3.1 Uplink Power Control . . . 35
5.3.2 Uplink Timing Control . . . 36
5.4 Logging . . . 36
5.5 The Command Line Interface (CLI) . . . 37
5.5.1 Attaching to the OpenBTS CLI . . . 37
5.5.2 “alarms” Command . . . 37
5.5.3 “calls” Command . . . 38
5.5.4 “cellid” Command . . . 38
5.5.5 “chans” Command . . . 38
5.5.6 “config” & “unconfig” Commands . . . 39
5.5.7 “endcall” Command . . . 39

5.5.8 "exit" Command . . . 40
5.5.9 "load" Command . . . 40
5.5.10 "notices" Command . . . 40
5.5.11 "page" Command . . . 40
5.5.12 "power" Command . . . 40
5.5.13 "sendsms" and "sendsimple" Commands . . . 41
5.5.14 "testcall" Command . . . 41
5.5.15 "tmsis" Command . . . 41
5.5.16 "version" Command . . . 41
5.5.17 CLI Shell Escape . . . 41
5.6 Open Registration . . . 42

6 Asterisk and the Subscriber Registry 43
6.1 Real Time Asterisk & the Subscriber Registry . . . 43
6.1.1 Configuring the Subscriber Registry . . . 43
6.2 Provisioning New Subscribers . . . 44
6.2.1 Using Pre-existing SIMs . . . 44
6.3 Emergency Calls . . . 45
6.3.1 What the User Dials . . . 45
6.3.2 Configuring OpenBTS to Support Emergency Calls . . . 46
6.4 Connecting to a VoIP Carrier . . . 46
6.5 Hybrid GSM/SIP Transactions . . . 47
6.5.1 Registration ("Location Updating") . . . 47
6.5.2 Call Control . . . 47

7 SMS Text Messaging 49
7.1 Internet Messaging Protocols . . . 49
7.1.1 The "Session" Problem . . . 49
7.1.2 RFC-3428 . . . 49
7.2 Smqueue . . . 50
7.2.1 Design and Operation of Smqueue . . . 50
7.2.2 Addressing in Smqueue . . . 50

7.2.3 Configuration of Smqueue 50
7.3 Text Messaging in GSM 51
7.3.1 Layers of SMS in OpenBTS and Smqueue 51
7.3.2 RFC-3428/SMS Transaction Ladders 52
7.4 Short Code Applications 53
7.4.1 Short Code Implementation 54
7.4.2 Existing Short Code Applications 54

8 Other GSM Services **56**
8.1 Radio Resource Location Protocol (RRLP) 56
8.1.1 The Need for Assistance 56
8.1.2 The RRLP Server 57
8.1.3 RRLP Service Controls and Configuration 58

List of Figures

2.1 Components of the OpenBTS application suite and their communication channels as installed in each access point. Sharp-cornered boxes are hardware components. Round-cornered boxes are software components. 16

3.1 Layers and channels of the Um interface. This figure shows the basic logical channel types in a subset of a typical configuration. 17

6.1 GSM location update mapped to a SIP REGISTER (non-authenticating case). 47

6.2 A GSM-SIP mobile-terminated call, VEA, normal case. 48

6.3 A GSM-SIP mobile-originated call, VEA, normal case. 48

7.1 Mobile-terminated SMS transfer with no parallel call, normal case. 53

7.2 Mobile-originated SMS transfer with no parallel call, normal case. 54

List of Tables

5.1 Maximum output power levels for GSM MSs. From GSM 05.05 Section 4.1.1. 35

Chapter 1

Introduction

The OpenBTS P2.8 "Opelousas" release is distributed publicly under the AGPLv3 license. P2.8 provides the following improvements over previous the P2.6 AGPLv3 "Mamou" public release:

- External configuration control and status reporting through an sqlite3 database file.
- Syslogd support.
- In-call delivery and submission of text messages.
- Support for binary payloads and non-Latin alphabets in SMS.
- RRLP.
- Standalone subscriber registry database and "realtime" Asterisk support.
- Many bug-fixes in the SIP and call control modules.
- Support for new types of digital radio hardware, beyond the Ettus USRP1:
  - Ettus USRP2
  - Range Networks RAD1 and Leapfrog devices

The P2.8 release is derived from the C2.8 commercial release.[1] The C2.8 release has the same architecture as P2.8, but includes the following additional features:

- Emergency call support, with priority channel allocation and optional IMS headers.
- SMSCB.
- Multi-ARFCN operation.
- A3/A8 authentication based on a SIP challenge-response exchange.

For more information about products based on commercial releases of OpenBTS, see Section 1.5.

[1] There is no P2.7 release; the C2.7 "Natchitoches" commercial release had no corresponding public release.

1.1 Scope and Audience

This document describes the organization and operation of the OpenBTS P2.8-series GSM access point software. It is intended for use by software developers.

1.2 Disclaimers

1.2.1 Warranty

The OpenBTS software and its associated documentation are provided with NO WARRANTY OF ANY KIND.

Although the information in this handbook has been carefully checked for accuracy, and is believed to be correct and current, no warranty, either express or implied, is made as to either its applicability to, or its compatibility with, specific requirements; nor does Range Networks, Inc. assume any responsibility for correctness of this information, nor for damages consequent to its use. All design characteristics and specifications are subject to change without notice.

Range Networks, Inc. welcomes any reports of software failures or documentation errors and will make reasonable efforts to correct these in future releases.

1.2.2 Accuracy

This manual is a description of the OpenBTS software, not a specification. In any discrepancy between the software and this manual, the software source code is authoritative.

1.2.3 Patent Laws

OpenBTS P2.8 is distributed for educational and experimental purposes. The use of this software to provide GSM services may result in the use of patented technologies. The user of this software is required to take whatever actions are necessary to avoid patent infringement.

1.2.4 Trademarks

"OpenBTS" is a registered trademark of Range Networks, Inc. (Range), a US corporation. Range reserves the right to control the use of this trademark. Do not use this trademark in commerce without permission and do not rebrand OpenBTS under a different trademark.

"Asterisk" is a trademark of Digium, Inc.

1.2.5 Telecom and Radio Spectrum Laws

Users of this software are expected to comply with all applicable local, national and international regulations.

1.2.6 Software Licensing and Distribution

OpenBTS P2.8 is distributed in source code form under APGLv3, protected under copyright law. This software is not "public domain". AGPLv3 is a commercial software license, enforceable like any other.

OpenBTS P2.8 also incorporates other components acquired under GPL-family licenses.

GPL Compliance

The following components of OpenBTS P2.8 were acquired under GPL variants and are included in source code form in the AGPLv3 distributions:

- URSP interface libraries use in the "transceiver" applciation

LGPL Compliance

The following publicly-available packages are linked dynamically by applications in the OpenBTS P2.8 suite under LGPL licenses:

- UnixODBC. Public distribution available from http://www.unixodbc.org/. License is LGPL v2.1.
- libosip2. Public distribution available from http://www.gnu.org/software/osip/. License is LGPL.
- libortp. Public distribution available from http://www.linphone.org/. License is LGPL v2.1.

In each case, OpenBTS uses the public distribution without modification.

1.3 Abbreviations

- APDU – application protocol data unit
- ARFCN – absolute radio frequency channel number
- BTS – base transceiver subsystem
- dB – Decibels
- dBm – Decibel milliwatts
- FEC – forward error correction
- GPL – General Public License
- kHz – kilohertz
- LNA – low-noise amplifier
- LGPL – Lesser General Public License
- MOC – mobile-originated call

- MO-SMS – mobile-originated SMS
- MTC – mobile-terminated call
- MT-SMS – mobile-terminated SMS
- MS – mobile station
- PA – power amplifier
- PDU – protocol data unit
- preamp – low-noise amplifier
- RF – radio frequency
- RPDU – relay-layer protocol data unit
- RRLP – radio resource location protocol
- SIP – Session Initiation Protocol
- SMS – Short Message Service
- TDM – time-division multiplexing
- TPDU – transfer-layer protocol data unit
- USSD – unstructured supplementary service data
- VDC – Volts, direct current
- W – Watts

1.4 References

1.4.1 ETSI/3GPP

This document references the following GSM and 3GPP specifications, which can be downloaded for free from

http://webapp.etsi.org/key/queryform.asp

- GSM 03.38 “Digital cellular telecommunications system (Phase 2+); Alphabets and language-specific information”
- GSM 03.41: “Digital cellular telecommunications system (Phase 2+); Technical realization of Cell Broadcast Service (CBS)”
- GSM 04.06: “Digital cellular telecommunications system (Phase 2+); Mobile Station - Base Station System (MS - BSS) interface; Data Link (DL) layer specification”

- GSM 04.08: "Digital cellular telecommunications system (Phase 2+); Mobile radio interface layer 3 specification"
- GSM 05.02: "Digital cellular telecommunications system (Phase 2+); Multiplexing and multiple access on the radio path"
- GSM 05.03: "Digital cellular telecommunications system (Phase 2+); Channel coding"
- GSM 05.05: "Digital cellular telecommunications system (Phase 2+); Radio transmission and reception"
- GSM 05.08: "Digital cellular telecommunications system (Phase 2+); Radio subsystem link control"
- GSM 05.10: "Digital cellular telecommunications system (Phase 2+); Radio subsystem synchronization"
- 3GPP 24.229: "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3"

1.4.2 Unix Documentation

This document references manual pages for the following Linux/Unix utilities:

- screen
- syslogd and syslogd.conf

1.4.3 IETF

This document references the following IETF standards, which can be downloaded for free from

http://tools.ietf.org/html/

- RFC-2833: "RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals"
- RFC-2976: "The SIP INFO Method"
- RFC-3261: "SIP: Session Initiation Protocol"
- RFC-3325: "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks"
- RFC-3428: "Session Initiation Protocol (SIP) Extension for Instant Messaging"
- RFC-3455: "Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3rd-Generation Partnership Project (3GPP)"
- RFC-3550: "RTP: A Transport Protocol for Real-Time Applications"
- RFC-4119: "A Presence-based GEOPRIV Location Object Format"

1.5 Contact Information

For information on Range Networks' commercial products and professional support for OpenBTS, please contact:

Range Networks, Inc.
560 Brannan Street
San Francisco, California 94107
United States of America

telephone +1 415-778-8700

email support@rangenetworks.com or sales@rangenetworks.com

Chapter 2

The OpenBTS Application Suite

A complete OpenBTS P2.8 installation comprises several distinct applications:

- **OpenBTS** – The actual OpenBTS application, containing most of the GSM stack above the radiomodem.
- **Transceiver** – The software radiomodem and hardware control interface.
- **Asterisk** – The VoIP PBX in the standard public release configuration.
- **Smqueue** – The RFC-3428 store-and-forward server for text messaging.
- **Subscriber Registry** – A database of subscriber information that replaces both the Asterisk SIP registry and the GSM Home Location Register (HLR).
- **Other Servers** – Other optional GSM services, beyond speech and text messaging, are supported through external servers, interfaced to OpenBTS through HTTP or HTTPS with the server implemented as a set of CGI programs in an HTTP server. In OpenBTS P2.8 these servers are:
  - RRLP, for accessing the GPS receiver in most newer MSs

  See Chapter 8 for specific information on these servers.

The OpenBTS and Transceiver applications must run inside each GSM/SIP access point. The Asterisk and the subscriber registry applications communicated through the filesystem and therefore must run on the same computer, but that computer can be remote from the access point. smqueue and the other servers can run anywhere and may have multiple instances.

2.1 OpenBTS

The OpenBTS application contains:

- L1 TDM functions (GSM 05.02)
- L1 FEC functions (GSM 05.03)

- L1 closed loop power and timing controls (GSM 05.08 and 05.10)
- L2 LAPDm (GSM 04.06)
- L3 radio resource management functions (GSM 04.08)
- L3 GSM-SIP gateway for mobility management
- L3 GSM-SIP gateway for call control
- L4 GSM-SIP gateway for text messaging

The general design approach of OpenBTS is avoid implementing any function above L3, so at L3 or L4 every subprotocol of GSM is either terminated locally or translated through a gateway to some other protocol for handling by an external application. Similarly, OpenBTS itself does not contain any speech transcoding functions above the L1 FEC parts.

2.2 Transceiver

The transceiver application performs the radiomodem functions of GSM 05.05 and manages the USB interface to the radio hardware. The functions of the transceiver are described in Section 3.1.1.

2.3 Asterisk

OpenBTS uses a SIP switch or PBX to perform the call control functions that would normally be performed by the mobile switching center in a conventional GSM network, although in most network configurations this switching function is distributed over multiple switches. These switches also provide transcoding services.

In OpenBTS P2.8 the standard SIP switch is Asterisk 1.8. For more information on Asterisk itself, a good resource is the book Asterisk: The Future of Telephony by Jim Van Meggelen, Jared Smith, and Leif Madsen, from O'Reilly Publishing, ISBN 0-596-00962-3. OpenBTS has been used with VoIP PBX applications other than Asterisk, however Range does not normally support those configurations.

See Chapter 6 for information about integration between OpenBTS and Asterisk.

2.4 Subscriber Registry

OpenBTS uses a modified SIP registry as a substitute for the home location register found in a conventional GSM network. OpenBTS also relies on Asterisk for any transcoding functions. See Chapter 6 for information about the subscriber registry and its integration with OpenBTS and Asterisk.

2.5 Smqueue

Smqueue is an RFC-3428 store-and-forward server that is used for text messaging in the OpenBTS system. Smqueue is required to send a text message from one MS to another, or to provide reliable delivery of text

messages to an MS from any source. See Chapter 7 for more information.

2.6 Network Organization

In the simplest network, with a single access point, all of the applications in the suite run inside the access point on the same embedded computer. This is shown in Figure 2.1.

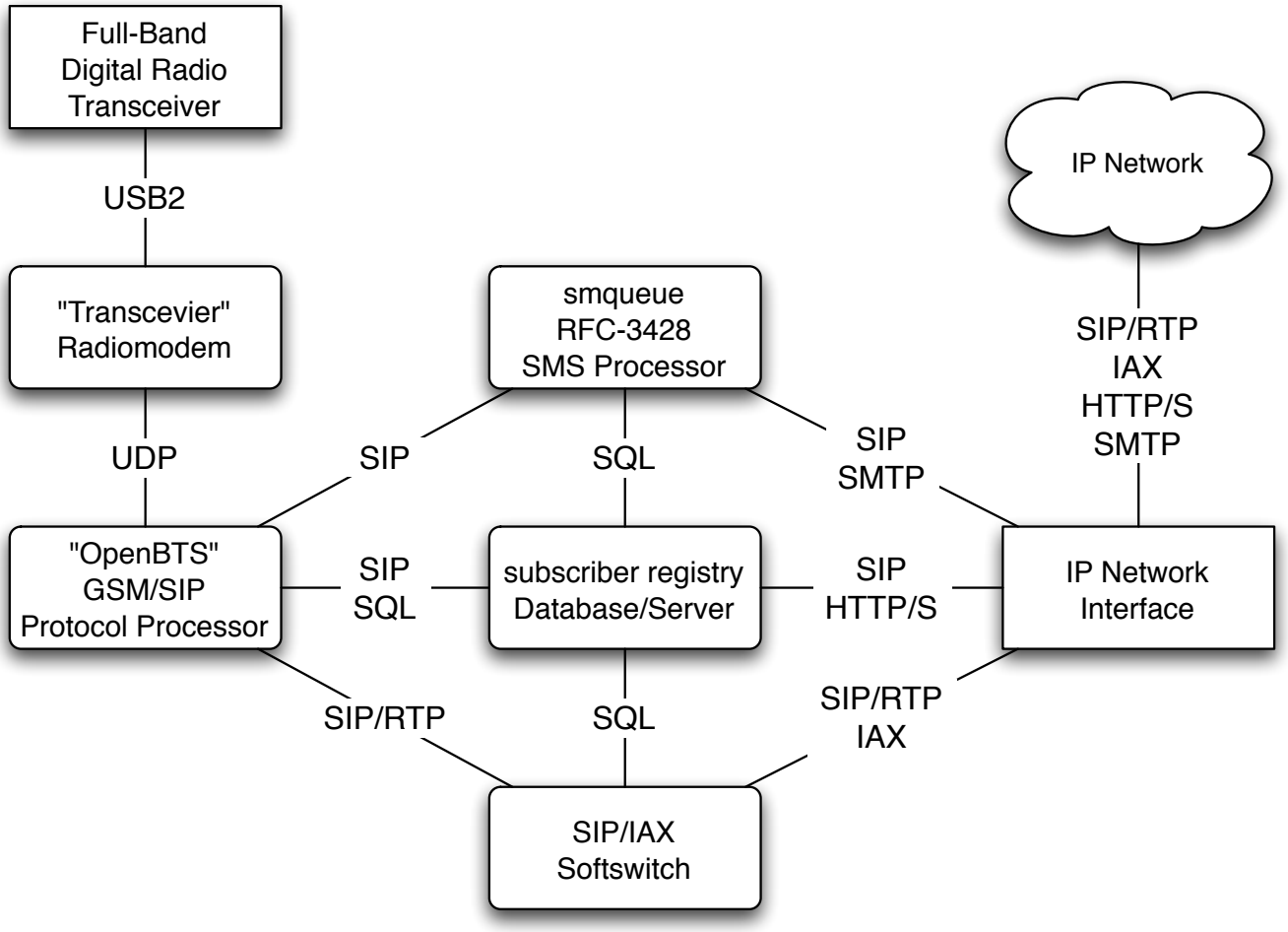

Figure 2.1: Components of the OpenBTS application suite and their communication channels as installed in each access point. Sharp-cornered boxes are hardware components. Round-cornered boxes are software components.

Chapter 3

The OpenBTS GSM Air Interface

This chapter describes the GSM air interface, "Um", as implemented by OpenBTS. It is not really necessary to fully understand this chapter to use OpenBTS, but the information is given here for completeness and to provide references to important parts of the GSM specifications to support more detailed study.

Broadly speaking, Um is organized into channels and layers, as shown in Figure 3.1. The rest of this chapter will explain this diagram.

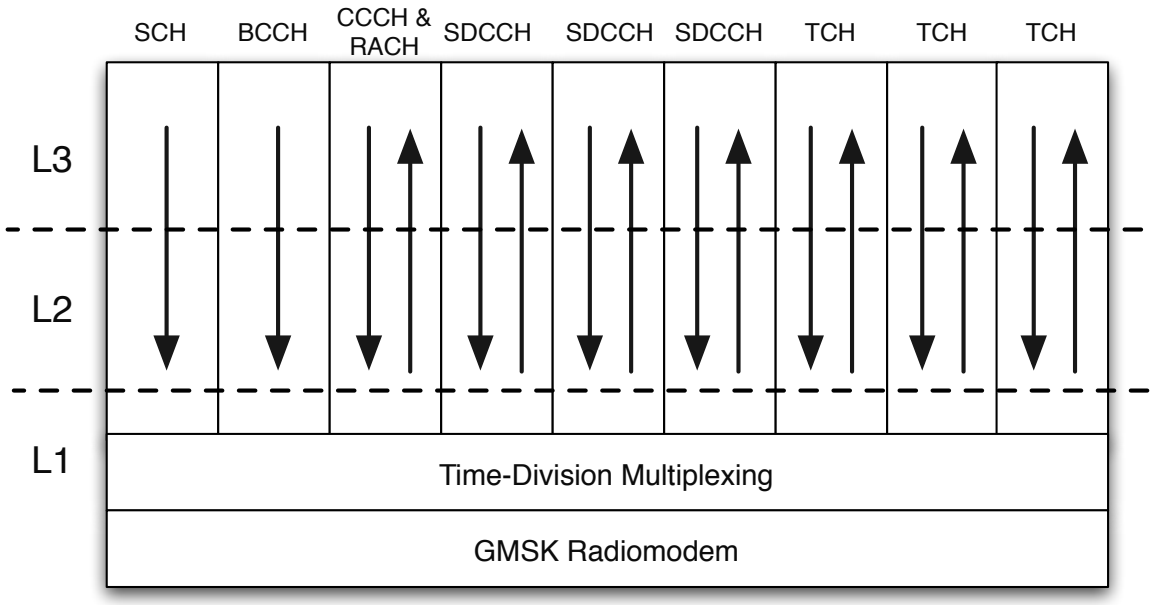

Figure 3.1: Layers and channels of the Um interface. This figure shows the basic logical channel types in a subset of a typical configuration.

3.1 Um Layers

The layers of GSM are initially defined in GSM 04.01 Section 7 and roughly follow the OSI model. Um is defined in the lower three layers of the model.

3.1.1 Physical Layer (L1)

The Um physical layer is defined in the GSM 05.xx series of specifications, with the introduction and overview in GSM 05.01. For most channels, Um L1 transmits and receives 184-bit control frames or 260-bit vocoder frames over the radio interface in 148-bit bursts with one burst per timeslot. There are three sublayers:

- Radiomodem. This is the actual radio transceiver, defined in largely in GSM 05.04 and 05.05.
- Multiplexing and Timing. GSM uses TDMA to subdivide each radio channel into as many as 16 traffic channels or as many as 64 control channels. The multiplexing patterns are defined in GSM 05.02.
- FEC Coding. This sublayer provides bit-error concealment and recovery. This sublayer is defined in GSM 05.03.

Radiomodem

OpenBTS P2.8 supports GMSK modulation with a 13/48 MHz (270.833 kHz) symbol rate and a channel spacing of 200 kHz. Since adjacent channels overlap, the standard does not allow adjacent channels to be used in the same cell. OpenBTS P2.8 supports the four most common GSM bands:

- GSM850, used in parts of ITU region 2
- PGSM900 and EGSM900, used in most of the world
- DCS1800, used in most of the world
- PCS1900, used in parts of ITU region 2

GSM is frequency duplexed, meaning that the network and MS transmit on different frequencies, allowing the BTS to transmit and receive at the same time. Transmission from the network to the MS is called "downlink". Transmission from the MS to the network is called "uplink". GSM uplink and downlink bands are separated by 45 or 50 MHz, depending on the specific band.

Uplink/downlink channel pairs are identified by an index called the ARFCN. Within the BTS, these ARFCNs are given arbitrary carrier indexes C0, C1, etc., with C0 designated as a Beacon Channel and always operated at constant power. The radio channel is time-multiplexed into 8 timeslots, each with a duration of 156.25 symbol periods. These 8 timeslots form a frame of 1,250 symbol periods. The capacity associated with a single timeslot on a single ARFCN is called a physical channel (PCH) and referred to as "CnTm" where n is a carrier index and m is a timeslot index (0-7).

Each timeslot is occupied by a radio burst with a guard interval, two payload fields, tail bits, and a midamble (or training sequence). The lengths of these fields vary with the burst type but the total burst

length is always 156.25 symbol periods. The most commonly used burst is the Normal Burst (NB). There are several other burst formats, though. Bursts that require higher processing gain for signal acquisition have longer midambles. The random access burst (RACH) has an extended guard period to allow it to be transmitted with incomplete timing acquisition. Burst formats are described in GSM 05.02 Section 5.2.

Multiplexing and Timing

Each physical channel is time-multiplexed into multiple logical channels according to the rules of GSM 05.02. Traffic channel multiplexing follows a 26-frame (0.12 second) cycle called a "multiframe". Control channels follow a 51-frame multiframe cycle. The C0T0 physical channel carries the SCH, which encodes the timing state of the BTS to facilitate synchronization to the TDMA pattern.

GSM timing is driven by the serving BTS through the SCH and FCCH. All clocks in the MS, including the symbol clock and local oscillator, are slaved to signals received from the BTS, as described in GSM 05.10. BTSs in the GSM network can be asynchronous, so that each BTS can run an independent clock.

FEC Coding

The coding sublayer provides forward error correction. As a general rule, each GSM channel uses a block parity code (usually a Fire code), a rate-1/2, 4th-order convolutional code and a 4-burst or 8-burst interleaver. Notable exceptions are the synchronization channel (SCH) and random access channel (RACH) that use single-burst transmissions and thus have no interleavers. For speech channels, vocoder bits are sorted into importance classes with different degrees of encoding protection applied to each class (GSM 05.03). Using soft-input Viterbi decoding, the FEC decoders in OpenBTS can recover frames reliably with bit erasure rates in excess of 25%.

Most channels in GSM use 456-bit L1 frames. On channels with 4-burst interleaving (BCCH, CCCH, SDCCH, SACCH), these 456 bits are interleaved in to 4 radio bursts with 114 payload bits per burst. On channels with 8-burst interleaving (TCH, FACCH), these 456 bits are interleaved over 8 radio bursts so that each radio burst carries 57 bits from the current L1 frame and 57 bits from the previous L1 frame. Interleaving algorithms for the most common traffic and control channels are described in GSM 05.03 Sections 3.1.3, 3.2.3 and 4.1.4.

3.1.2 Data Link Layer (L2)

The Um data link layer, LAPDm, is defined in GSM 04.05 and 04.06. LAPDm is the mobile analog to ISDN's LAPD and like LAPD, LAPDm is a simplified form of HDLC.

3.1.3 Network Layer (L3)

Um L3 is defined in GSM 04.07 and 04.08 and has three sublayers. A subscriber terminal must establish a connection in each sublayer before accessing the next higher sublayer.

- Radio Resource (RR). This sublayer manages the assignment and release of logical channels on the radio link. It is normally terminated in the BSC, although in OpenBTS, RR is terminated locally in the OpenBTS stack.

- Mobility Management (MM). This sublayer authenticates users and tracks their movements from cell to cell. OpenBTS translates MM transactions into corresponding SIP transactions and uses the Asterisk SIP registry to perform MM functions.
- Call Control (CC). This sublayer connects telephone calls and is taken directly from ITU-T Q.931. GSM 04.08 Annex E provides a table of corresponding paragraphs in GSM 04.08 and ITU-T Q.931 along with a summary of differences between the two. In OpenBTS, CC transactions are translated to corresponding SIP transactions and processed in Asterisk.

The access order is RR, MM, CC. The release order is the reverse of that.

3.2 Um logical channels

Um logical channel types are outlined in GSM 04.03. Broadly speaking, non-GRPS Um logical channels fall into three categories: traffic channels, dedicated control channels and non-dedicated control channels.

3.2.1 Traffic channels (TCH)

These point-to-point channels correspond to the ISDN B channel and are referred to as Bm channels. Traffic channels use 8-burst diagonal interleaving with a new block starting on every fourth burst and any given burst containing bits from two different traffic frames. This interleaving pattern makes the TCH robust against single-burst fades since the loss of a single burst destroys only 1/8 of the frame's channel bits (a 12.5% bit erasure). The coding of a traffic channel is dependent on the traffic or vocoder type employed, with most coders capable of overcoming single-burst losses. All traffic channels use a 26-multiframe TDMA structure.

Full-rate channels (TCH/F)

A GSM full rate channel uses 24 frames out of a 26-multiframe. The channel bit rate of a full-rate GSM channel is 22.7 kbit/s, although the actual payload data rate is 9.6-14 kbit/s, depending on the channel coding. OpenBTS P2.8 supports only the GSM full-rate codec (GSM 06.10) as a media type on this channel.

3.2.2 Dedicated Control Channels (DCCHs)

These point-to-point channels correspond to the ISDN D channel and are referred to as Dm channels.

Standalone Dedicated Control Channel (SDCCH)

The SDCCH is used for most short transactions, including initial call setup step, registration and SMS transfer. It has a payload data rate of 0.8 kbit/s. Up to eight SDCCHs can be time-multiplexed onto a single physical channel. The SDCCH uses 4-burst block interleaving in a 51-multiframe. One SDCCH channel can be used to process 10-15 location updates per minute or to transfer 5-10 SMS per minute.

Fast Associated Control Channel (FACCH)

The FACCH is always paired with a traffic channel. The FACCH is a blank-and-burst channel that operates by stealing bursts from its associated traffic channel. Bursts that carry FACCH data are distinguished from traffic bursts by stealing bits at each end of the midamble. The FACCH is used for in-call signaling, including call disconnect, handover and the later stages of call setup. It has a payload data rate of 9.2 kbit/s when paired with a full-rate channel (FACCH/F) and 4.6 kbit/s when paired with a half-rate channel (FACCH/H). The FACCH uses the same interleaving and multiframe structure as its host TCH.

Slow Associated Control Channel (SACCH)

Every SDCCH or FACCH also has an associated SACCH. Its normal function is to carry system information messages 5 and 6 on the downlink, carry receiver measurement reports on the uplink and to perform closed-loop power and timing control. Closed loop timing and power control are performed with a physical header at the start of each L1 frame. This 16-bit physical header carries actual power and timing advance settings in the uplink and ordered power and timing values in the downlink. The SACCH can also be used for in-call delivery of SMS. The SACCH has a payload data rate of 0.2-0.4 kbit/s, depending on the channel with which it is associated. The SACCH uses 4-burst block interleaving and the same multiframe type as its host TCH or SDCCH.

3.2.3 Non-Dedicated Control Channels (NDCCHs)

These are unicast and broadcast channels that do not have analogs in ISDN. These channels are used almost exclusively for radio resource management. The CCCH and RACH together form the medium access mechanism for Um.

Broadcast Control Channel (BCCH)

The BCCH carries a repeating pattern of system information messages that describe the identity, configuration and available features of the BTS. BCCH brings the measurement reports it bring the information about LAI And CGI BCCH frequency are fixed in BTS. The C0T0 beacon channel must carry an instance of the BCCH.

Synchronization Channel (SCH)

The SCH transmits a Base station identity code and the current value of the TDMA clock. The C0T0 beacon channel must carry an instance of the SCH.

Frequency Correction Channel (FCCH)

The FCCH generates a tone on the radio channel that is used by the MS to discipline its local oscillator.

Common Control Channel (CCCH)

The CCCH is a downlink unicast channel that carries paging requests and channel assignment messages (specifically, immediate assignment messages). The CCCH is subdivided into the paging channel (PCH) and access grant channel (AGCH). An MS that is camped to a BTS monitors the PCH for service notifications from the network.

Random Access Channel (RACH)

The RACH is the uplink counterpart to the CCCH. The RACH is a shared channel on which the MSs transmit random access bursts to request channel assignments from the BTS, assignments which are granted on the AGCH part of the CCCH.

3.2.4 Allowed channel combinations

The multiplexing rules of GSM 05.02 allow only certain combinations of logical channels to share a physical channel. The combinations supported by OpenBTS P2.8 are:

- Combination I: TCH/F + FACCH/F + SACCH. This combination is used for full rate traffic. It can be used anywhere but C0T0.
- Combination V: FCCH + SCH + BCCH + CCCH + 4 SDCCH + 4 SACCH. This is the typical C0T0 beacon channel combination for small cells. It can be used only on C0T0. Since this is the only beacon channel combination supported by OpenBTS P2.8, it *must* be used on C0T0.
- Combination VII: 8 SDCCH + 8 SACCH. This combination is used to provide additional SDCCH capacity in situations where registration loads or SMS usage may be particularly heavy. It can be used anywhere but C0T0.

3.3 Service Capacity of Um

The capacities of OpenBTS products, ARFCN-for-ARFCN, are the same as for any other GSM basesations. The only exception to this is that OepnBTS P2.8 doe not support half-rate channels.

OpenBTS P2.8 offers two types of dedicated channels:

- Full-Rate Traffic Channel (TCH/F). Each Combination-I slot contains a single TCH/F that can carry a single speech call.
- Standalone Dedicated Control Channel (SDCCH). Each Combination-VII slot carries eight SDCCHs. The Combination-V beacon also carries four SDCCHs. Each SDCCH can process about 30 authenticated registration transactions per minute or transfer about 12 text messages per minute, assuming good link margins. *Bear in mind that poor link margins will significantly degrade SDCCH capacity by forcing retransmission of L2 frames and requiring long tear-down times for dropped channels.*

A typical configuration for a single-ARFCN BTS in a speech-oriented application is

- a Combination-IV beacon on C0T0 carrying 4 SDCCHs and
- six Combination-I slots on C0T1-C0T7 carrying total of 7 TCH/Fs.

This combination would typically support about about 15 authenticated registrations per minute, about 40 text messages per minute and seven concurrent calls. The number of subscribers that can actually be served with that capacity will be covered in the following sections.

Chapter 4

External Databases

OpenBTS P2.8 uses a set of sqlite3 database files to make its configuration and status information available to external applications. Sqlite3 is a self-contained, serverless SQL database, originally developed for guided missile systems and now used many well-known applications, including the Blackberry, Symbian, iOS and Android operating systems. For more information on sqlite3, see the www.sqlite.org web site.

4.1 Editing Sqlite3 Databases

The following methods can be used to edit or view the OpenBTS sqlite3 database files:

- The sqlite3 command line tool. Range Networks access points include the sqlite3 command line tool that can be used to inspect and modify these databases using SQL syntax.
  - The database can be manipulated directly using SQL syntax in real time.
  - For offline editing, sqlite3 can export SQL code to a text file with ".dump". The text can then be edited and reimported with ".read".
- Third-party database editors. Generic GUI-based editors are available for sqlite3 database files. Examples:
  - "SQLite Database Browser" – A free browser for OS X and Linux available from sourceforge.net.
  - "RazorSQL" – A commercial database GUI available from razorsql.com.
  - Firefox – Mozilla offers a free Firefox add-on called SQLite Manager that allows the Firefox web browser to be used to view and edit sqlite3 databases, available from addons.mozilla.org.
- OpenBTS itself. The OpenBTS CLI "config" and "unconfig" commands (Section 5.5) can be used to edit the configuration table (Section 4.2) in real time. Configuration changes from the CLI are written back the to OpenBTS.db database and are persistent.

4.2 The Configuration Table

The parameters that control the OpenBTS application are stored in a database table called the *configuration table*. Some parameters are *dynamic*, meaning that a parameter change will have an immediate effec.

Some of these parameters are *static* and changes to them do not take effect until OpenBTS is restarted. Some of these static parameters are matched to the hardware of a specific implementation and should not be changed at all. Comments within the configuration database describe each parameter and under what conditions it can be changed. Flags within the database schema indicate which parameters are static. The schema for the P2.8 configuration table is:

```
CREATE TABLE CONFIG (
    KEYSTRING TEXT UNIQUE NOT NULL,
    VALUESTRING TEXT,
    STATIC INTEGER DEFAULT 0,
    OPTIONAL INTEGER DEFAULT 0,
    COMMENTS TEXT DEFAULT ''
)
```

Note that the database itself contains comments, available to the operator at all times and repeated in this manual.

To change a dynamic configuration parameter in real time, edit the table using one of the methods described in Section 4.1. The effect will be immediate, although in-progress transactions may continue to use the parameters with which they started. (For example, a change in SIP.Proxy.Speech will not affect in-progress telephone calls, but any new calls will use the new proxy.)

To change a static configuration parameter. edit the table using one of the methods described in Section 4.1 and then restart the OpenBTS application. In standard-configuration embedded Linux systems, this restart can be accomplished with the "exit" command, described in Section 5.5.8 or by rebooting the BTS unit.

In some cases, NULL is a valid value for a configuration parameter. In SQL terminology, the NULL value is a special tag that indicates that a value is not defined. This is not the same as the string "NULL", which is an actual string value with a value of "NULL". To make a value NULL from the CLI, use the "unconfig" command.

The configuration parameters are listed here briefly, but some parameters of particular importance are covered in later sections.

- CLI.Prompt – Prompt for the OpenBTS command line interface.
- Control.Emergency.Destination.Host – SIP destination host to be used for the "To:" header of emergency calls. This host may be different from the address given for SIP.Proxy.Emergency.
- Control.Emergency.Destination.User – SIP destination user or extension to be used for the "To:" header of emegency calls. IMS specifies "sos", but correct value must be matched to your switch configuration and PSAP interface.
- Control.Emergency.GatewaySwitch – Gateway SIP switch for inbound calls from other networks. This host is used to form the return path for emergency calls, so it should be a host address that will route from your serving PSAP.
- Control.Emergency.Geolocation – If defined, send this location as an RFC-4119 XML GEOPRIV object during SIP emergency call establishment. Format is dd:mm:ss[NS] ddd:dd:dd[EW].
- Control.Emergency.QueueTime – Maximum time to wait for a channel to open up for an emegency call in a congested system, in milliseconds.

- Control.Emergency.RFC5031 – If not NULL, use the RFC-5031 URN sip:sos@SIP.Proxy.Emergency as the request URN for outbound emergency calls over SIP, regardless of the value of Emergency.Destination.User. The "To:" header will still be Emergency.Destination.User@Emergency.Destination.Host.
- Control.Emergency.Source.User – SIP identity to use if no IMSI is available. IMS specifies "anonymous" but other values might be more useful depending on your configuration.
- Control.GSMTAP.TargetIP – Target IP address for GSMTAP packets; the IP address of Wireshark, if you use it for GSM.
- Control.LUR.AttachDetach – Attach/detach flag. Set to 1 to use attach/detach procedure, 0 otherwise. This will make initial LUR more prompt. It will also cause an un-regstration if the handset powers off and really heavy LUR loads in areas with spotty coverage.
- Control.LUR.CachedAuthentication – If not NULL, use RAND-SRES pairs cached in the TMSI table for authentication. If this method is used, it is important that the TMSI Table be in persistent storage.
- Control.LUR.DefaultAuthenticationAccept – If not NULL, provisioned handsets will be accepted for authentication in the absence of any defined authentication method. For commercial networks this value should probably be NULL.
- Control.LUR.FailedRegistration.Message – If defined, send this text message, followed by the IMSI, to unprovisioned handsets that are denied registration.
- Control.LUR.FailedRegistration.ShortCode – The return address for the failed registration message. If the message is defined, this must also be defined.
- Control.LUR.NormalRegistration.Message – If defined, send this text message, followed by the IMSI, to provisioned handsets when they attach on Um.
- Control.LUR.NormalRegistration.ShortCode – The return address for the normal registration message. If the message is defined, this must also be defined.
- Control.LUR.OpenRegistration – A regular expression. If not NULL, allow unprovisioned handsets with matching IMSIs to attach in Um.
- Control.LUR.OpenRegistration.Message – If defined, send this text message, followed by the IMSI, to unprovisioned handsets when they attach on Um due to open registration.
- Control.LUR.OpenRegistration.ShortCode – The return address for the open registration message. If the message is defined, this must also be defined.
- Control.LUR.QueryClassmark – If not NULL, query every MS for classmark during LUR.
- Control.LUR.QueryIMEI – If not NULL, query every MS for IMSI during LUR.
- Control.LUR.SR-HTTPAuthentication – If not NULL, use the HTTP interface of the subscriber registry server for authentication.
- Control.LUR.SendTMSIs – If not NULL, send new TMSI assignments to handsets that are allowed to attach.

- Control.LUR.UnprovisionedRejectCause – Reject cause for location updating failures for unprovisioned phones. Reject causes come from GSM 04.08 10.5.3.6. Reject cause 0x04, IMSI not in VLR, is usually the right one.
- Control.NumSQLTries – Number of times to retry SQL queries before declaring a database access failure.
- Control.Reporting.PhysStatusTable – File path for channel status reporting database. Static.
- Control.Reporting.TMSITable – File path for TMSITable database. Static.
- Control.TMSITable.MaxAge – Maximum allowed age for a TMSI in hours.
- Control.TMSITable.MaxSize – Maximum size of TMSI table before oldest TMSIs are discarded.
- Control.TestCall.Port – Port for exchanging L3 packets with the testcall feature.
- Control.VEA – If not NULL, user very early assignment for speech call establishment. See GSM 04.08 Section 7.3.2 for a detailed explanation of assignment types. If VEA is selection, GSM.CellSelection.NECI should be set to 1. See GSM 04.08 Sections 9.1.8 and 10.5.2.4 for an explanation of the NECI bit.
- GSM.CCCH.AGCH.QMax – Maximum number of access grants to be queued for transmission on AGCH before declaring congrestion.
- GSM.CCCH.CCCH-CONF – CCCH configuration type. See GSM 10.5.2.11 for encoding. Value of 1 means we are using a C-V beacon. Any other value selects a C-IV beacon.
- GSM.CCCH.PCH.Reserve – Number of CCCH subchannels to reserve for paging.
- GSM.CellSelection.CELL-RESELECT-HYSTERESIS – Cell Reselection Hysteresis. See GSM 04.08 10.5.2.4, Table 10.5.23 for encoding. Encoding is $2N$ dB, values of N are 0...7 for 0...14 dB.
- GSM.CellSelection.MS-TXPWR-MAX-CCH – Cell selection parameters. See GSM 04.08 10.5.2.4.
- GSM.CellSelection.NCCsPermitted – NCCs Permitted. An 8-bit mask of allowed NCCs. Unless you are coordinating with another carrier, this should probably just select your own NCC.
- GSM.CellSelection.NECI – NECI, New Establishment Causes. This must be set to "1" if you want to support very early assignment. See GSM 04.08 10.5.2.4, Table 10.5.23 and 04.08 9.1.8, Table 9.9.
- GSM.CellSelection.Neighbors – ARFCNs of neighboring cells.
- GSM.CellSelection.RXLEV-ACCESS-MIN – Cell selection parameters. See GSM 04.08 10.5.2.4.
- GSM.Channels.C1sFirst – If not NULL, allocate C-I slots first, starting at C0T1. Otherwise, allocate C-VII slots first. Static.
- GSM.Channels.NumC1s – Number of Combination-I timeslots to configure. The C-I slot carries a single full-rate TCH, used for speech calling. Static.
- GSM.Channels.NumC7s – Number of Combination-VII timeslots to configure. The C-VII slot carries 8 SDCCHs, useful to handle high registration loads or SMS. If C0T0 is C-IV, you must have at least one C-VII also. Static.

- GSM.Identity.BSIC.BCC – GSM basestation color code; lower 3 bits of the BSIC. BCC values in a multi-BTS network should be assigned so that BTS units with overlapping coverage do not share a BCC. This value will also select the training sequence used for all slots on this unit.
- GSM.Identity.BSIC.NCC – GSM network color code; upper 3 bits of the BSIC. Assigned by your national regulator. Must be distinct from NCCs of other GSM operators in your area.
- GSM.Identity.CI – Cell ID, 16 bits. Should be unique.
- GSM.Identity.LAC – Location area code, 16 bits, values 0xFFxx are reserved. For multi-BTS networks, assign a unique LAC to each BTS unit. (That is not the normal procedure in conventional GSM networks, but is the correct procedure in OpenBTS networks.)
- GSM.Identity.MCC – Mobile country code, 2 or 3 digits. Defined in ITU-T E.212.
- GSM.Identity.MNC – Mobile network code; Must be 3 dgits. Assigned by your national regulator.
- GSM.Identity.ShortName – Network short name, displayed on some phones. Optional but must be defined if you also want the network to send time-of-day.
- GSM.Identity.ShowCountry – If not NULL, tell the phone to show the country name based on the MCC.
- GSM.MS.Power.Damping – Damping value for MS power control loop.
- GSM.MS.Power.Max – Maximum commanded MS power level in dBm.
- GSM.MS.Power.Min – Minimum commanded MS power level in dBm.
- GSM.MS.TA.Damping – Damping value for timing advance control loop.
- GSM.MS.TA.Max – Maximum allowed timing advance in symbol periods. Ignore RACH bursts with delays greater than this. Can be used to limit service range.
- GSM.MaxSpeechLatency – Maximum allowed speech buffering latency, in 20 ms frames. If the jitter is larger than this delay, frames will be lost.
- GSM.RACH.AC – Access class flags. This is the raw parameter sent on the BCCH. See GSM 04.08 10.5.2.29 for encoding. Set to 0 to allow full access. If you do not have proper PSAP integration, set to 0x0400 to indicate no support for emergency calls.
- GSM.RACH.MaxRetrans – Maximum RACH retransmission attempts. This is the raw parameter sent on the BCCH. See GSM 04.08 10.5.2.29 for encoding.
- GSM.RACH.TxInteger – Parameter to spread RACH busts over time. This is the raw parameter sent on the BCCH. See GSM 04.08 10.5.2.29 for encoding.
- GSM.RADIO-LINK-TIMEOUT – L1 radio link timeout. This is the raw parameter sent on the BCCH; see GSM 10.5.2.3 for encoding. Should be coordinated with T3109.
- GSM.RRLP.ACCURACY – Requested accuracy of location request. K in 10(1.1**K-1). See 3GPP 03.32, sect 6.2
- GSM.RRLP.ALMANAC.REFRESH.TIME – How often the almanac is refreshed, in hours

- GSM.RRLP.ALMANAC.URL – URL of almanac source.
- GSM.RRLP.EPHEMERIS.REFRESH.TIME – How often the ephemeris is refreshed, in hours.
- GSM.RRLP.EPHEMERIS.URL – URL of ephemeris source.
- GSM.RRLP.RESPONSETIME – Mobile timeout. (OpenBTS timeout is 130 sec = max response time + 2.) N in 2**N. See 3GPP 04.31 sect A.2.2.1
- GSM.RRLP.SEED.ALTITUDE – Seed altitude in meters wrt geoidal surface.
- GSM.RRLP.SEED.LATITUDE – Seed latitude in degrees. -90 (south pole) .. +90 (north pole)
- GSM.RRLP.SEED.LONGITUDE – Seed longitude in degrees. -180 (west of greenwich) .. 180 (east)
- GSM.RRLP.SERVER.URL – URL of RRLP server.
- GSM.Radio.Band – The GSM operating band. Valid values are 850 (GSM850), 900 (PGSM900), 1800 (DCS1800) and 1900 (PCS1900). For most Range models, this value is dictated by the hardware and should not be changed. Static.
- GSM.Radio.C0 – The C0 ARFCN. Static.
- GSM.Radio.MaxExpectedDelaySpread – Expected worst-case delay spread in symbol periods, roughly 3.7 us or 1.1 km per unit.
- GSM.Radio.PowerManager.MaxAttenDB – Maximum transmitter attenuation level, in dB wrt full scale on the D/A output. This sets the minimum power output level in the output power control loop.
- GSM.Radio.PowerManager.MinAttenDB – Minimum transmitter attenuation level, in dB wrt full scale on the D/A output. This sets the maximum power output level in the output power control loop.
- GSM.Radio.PowerManager.NumSamples – Number of samples averaged by the output power control loop.
- GSM.Radio.PowerManager.SamplePeriod – Sample period for the output power control loop.
- GSM.Radio.PowerManager.TargetT3122 – Target value for T3122, the random access hold-off timer, for the power control loop.
- GSM.Radio.RSSITarget – Target uplink RSSI for MS power control loop, in dB wrt to A/D full scale. Should be 6-10 dB above the noise floor.
- GSM.Radio.RxGain – Receiver gain setting in dB. Ideal value is dictacted by the hardware. This database parameter is static but the receiver gain can be modified in real time with the CLI rxgain command. Static.
- GSM.Timer.T3113 – Paging timer T3113 in ms. This is the timeout for a handset to respond to a paging request. This should usually be the same as SIP.Timer.B in your VoIP network.
- GSM.Timer.T3122Max – Maximum allowed value for T3122, the RACH holdoff timer, in milliseconds.
- GSM.Timer.T3122Min – Minimum allowed value for T3122, the RACH holdoff timer, in milliseconds.

- GSM.Timer.T3212 – Registration timer T3212 period in minutes. Should be a factor of 6. Set to 0 to disable periodic registration. Should be smaller than SIP registration period.
- Log.Alarms.Max – Maximum number of alarms to remember inside the application.
- Log.Level – Default logging level when no other level is defined for a file.
- Log.Level.CallControl.cpp – Default configuration logs a trace at L3.
- Log.Level.MobilityManagement.cpp – Default configuration logs a trace at L3.
- Log.Level.RadioResource.cpp – Default configuration logs a trace at L3.
- Log.Level.SMSControl.cpp – Default configuration logs a trace at L3.
- NTP.Server – NTP server(s) for time-of-day clock syncing. For multiple servers, use a space-delimited list. If left undefined, NTP will not be used, but it is strongly recommended.
- RTP.Range – Range of RTP port pool. Pool is RTP.Start to RTP.Range-1. Static.
- RTP.Start – Base of RTP port pool. Pool is RTP.Start to RTP.Range-1. Static.
- SIP.DTMF.RFC2833 – If not NULL, use RFC-2833 (RTP event signalling) for in-call DTMF.
- SIP.DTMF.RFC2833.PayloadType – Payload type to use for RFC-2833 telephone event packets. If SIP.DTMF.2833 is defined, this must also be defined.
- SIP.DTMF.RFC2967 – If not NULL, use RFC-2967 (SIP INFO method) for in-call DTMF.
- SIP.Local.IP – IP address of the OpenBTS machine as seen by its proxies. If these are all local, this can be localhost. Static.
- SIP.Local.Port – IP port that OpenBTS uses for its SIP interface. Static.
- SIP.MaxForwards – Maximum allowed number of referrals.
- SIP.Proxy.Emergency – The IP host and port of the proxy to be used for emergency calls.
- SIP.Proxy.Registration – The IP host and port of the proxy to be used for registration and authentication. This is the subscriber registry, for example.
- SIP.Proxy.SMS – The IP host and port of the proxy to be used for text messaging. This is smqueue, for example.
- SIP.Proxy.Speech – The IP host and port of the proxy to be used for normal speech calls. This is Asterisk, for example.
- SIP.RegistrationPeriod – Registration period in minutes for MS SIP users. Should be longer than GSM T3212.
- SIP.SMSC – The SMSC handler in smqueue. This is the entity that handles full 3GPP MIME-encapsulted TPDUs. If not defined, use direct numeric addressing. Normally the value is NULL if SMS.MIMIEType is "text/plain" or "smsc" if SMS.MIMEType is "application/vnd.3gpp".
- SIP.Timer.A – INVITE retransmit period in ms.

- SIP.Timer.B – INVITE transaction timeout in ms. This value should usually match GSM.Timer.T3113.
- SIP.Timer.E – Non-INVITE initial request retransmit period in ms.
- SIP.Timer.F – Non-INVITE initial request timeout in ms.
- SIP.Timer.H – ACK timeout period in ms.
- SIP.Timer.I – ACK retransmit period in ms.
- SIP.Timer.J – Non-INVITE non-initial request retransmit period in ms.
- SIP.myPort – Port used by the SIP Authentication Server
- SMS.DefaultDestSMSC – Use this to fill in L4 SMSC address in SMS submission.
- SMS.FakeSrcSMSC – Use this to fill in L4 SMSC address in SMS delivery.
- SMS.MIMEType – This is the MIME Type that OpenBTS will use for RFC-3428 SIP MESSAGE payloads. Valid values are "application/vnd.3gpp.sms" and "text/plain".
- SubscriberRegistry.A3A8 – URL of upstream subscriber registry server. Blank (not NULL) if there is none.
- SubscriberRegistry.HTTP.Server – URL of the subscriber registry server.
- SubscriberRegistry.Manager.Title – Title of subscriber registry database manager web page.
- SubscriberRegistry.Manager.Url – URL of the subscriber registry database manager.
- SubscriberRegistry.Manager.VisibleColumns – Field names in subscriber registry visible in the database manager.
- SubscriberRegistry.db – The location of the sqlite3 database holding the subscriber registry.
- TRX.IP – IP address of the transceiver application. Static.
- TRX.Port – IP port of the transceiver application. Static.
- TRX.RadioFrequencyOffset – Fine-tuning adjustment for the transceiver. Static.

4.3 TMSI Table

To reduce dependence on a backhaul link, OpenBTS tracks TMSIs internally. To accomplish this, OpenBTS tracks TMSI-IMSI relationships in an sqlite3 database table called the *TMSI table*. TMSIs are assigned by a counter in increasing order. OpenBTS allocates a TMSI in the TMSI table for *every* MS that sends a Location Updating Request, whether the MS is allowed to register or not.

The TMSI table is treated as read-write by OpenBTS but should be treated as read-only by other applications. The path of the database file used for this table is defined in the configuration parameter Control.DBPath.TMSITable. In flash-based systems, this table should be stored in a ramdisk partition and its standard location is in /var/run. The schema is:

```
CREATE TABLE IF NOT EXISTS TMSI_TABLE (
  TMSI INTEGER PRIMARY KEY AUTOINCREMENT -- this value is used as the TMSI
  CREATED INTEGER NOT NULL, -- Unix time of record creation
  ACCESSED INTEGER NOT NULL,  -- Unix time of last encounter
  IMSI TEXT UNIQUE NOT NULL, -- IMSI of the SIM
  IMEI TEXT, -- IMEI of the MS, if requested
  L3TI INTEGER DEFAULT 0, -- L3 transaction identifier last used with this MS
  A5_SUPPORT INTEGER, -- encryption support in the MS, if requested
  POWER_CLASS INTEGER,  -- power class of the MS. if requested
  OLD_TMSI INTEGER,  -- previous TMSI from another cell or network
  PREV_MCC INTEGER, -- previous network MCC
  PREV_MNC INTEGER,  -- previous network MNC
  PREV_LAC INTEGER, -- previous network LAC
  RANDUPPER INTEGER -- cached authentication token
  RANDLOWER INTEGER -- cached authentication token
  SRES INTEGER, -- cached authentication token
  DEG_LAT FLOAT, -- cached RRLP result
  DEG_LONG FLOAT -- cached RRLP result
)
```

4.4 Channel Table

OpenBTS reports real-time physical status information for active dedicated channels to an external sqlite3 database table called PHYSTATUS. This table is treated as write-only by OpenBTS but should be treated as read-only by other applications. The entry for a channel is updated every time a system information message is received on the channel's associated SACCH. The path of the database file used for this table is defined in the configuration parameter GSM.DBPath.PhysStatusTable. The schema is:

```
CREATE TABLE IF NOT EXISTS PHYSTATUS (
  CN_TN_TYPE_AND_OFFSET STRING PRIMARY KEY,  -- cross-refs TRANSACTION_TABLE
  ARFCN INTEGER DEFAULT NULL, -- actual ARFCN
  ACCESSED INTEGER DEFAULT 0,  -- Unix time of last update
  RXLEV_FULL_SERVING_CELL INTEGER DEFAULT NULL,  -- from most recent measurement report
  RXLEV_SUB_SERVING_CELL INTEGER DEFAULT NULL,   -- from most recent measurement report
  RXQUAL_FULL_SERVING_CELL_BER FLOAT DEFAULT NULL, -- from most recent measurement report
  RXQUAL_SUB_SERVING_CELL_BER FLOAT DEFAULT NULL, -- from most recent measurement report
  RSSI FLOAT DEFAULT NULL, -- RSSI relative to full scale input
  TIME_ERR FLOAT DEFAULT NULL,  -- timing advance error in symbol periods
  TRANS_PWR INTEGER DEFAULT NULL, -- MS tx power in dBm
  TIME_ADVC INTEGER DEFAULT NULL, -- MS timing advance in symbol periods
  FER FLOAT DEFAULT NULL -- uplink FER
)
```

The CN_TN_TYPE_AND_OFFSET field is a channel description string of the form

```
C<n>T<n> <channelType>-<subchannelIndex>
```

For example

- "C0T1 TCH/F" is a full rate traffic channel on timeslot 1 of the C0 ARFCN and
- "C0T0 SDCCH-0/4" is the #0 SDCCH (of 4 available) on C0T0.

Strings of the same format are used in the transaction table.

Chapter 5

Processes Inside OpenBTS

Along with the fundamental channels and layers described in Chapter 3, OpenBTS P2.8 contains several processes that support its operation and tie the channels together to forms a unified BTS.[1]

5.1 T3122 Exponential Back-Off

When too many MSs make simultaneous access attempts to the BTS, resulting in channel exhaustion, the BTS can respond on the CCCH with an Immediate Assignment Reject message, as defined in GSM 04.08 Section 9.1.20. This message carries a value, T3122, that dictates how long the rejected MS must wait before making another access attempt. (Emergency call attempts are not subject to T3122 waiting.)

OpenBTS implements an exponential back-off algorithm that causes T3122 to grow exponentially whenever channel exhaustion occurs. The bounds for T3122 are set with the configuration parameters GSM.Timer.T3122Max and GSM.Timer.T3122Min, given in milliseconds. To disable the exponential back-off, set these two bounds to the same value.

T3122 back-off is connected to downlink power adaptation, described in Section 5.2.

5.2 Downlink Power and Congestion Management

OpenBTS can automatically adjust its downlink power to limit loads and prevent congestion. This feature is especially useful for graceful power-up in areas with very high subscriber density and load-shedding in the event of sudden failure of a neighboring cell or even the failure of a nearby cell of a different operator. This congestion management feature works in conjunction with the T3122 adaptation loop described in Section 5.1. The practical result of the automatic power adjustment is to limit the service area of the BTS to a population of nearby phones that it can actually serve.

The configuration parameters associated with this mechanism are:

- GSM.PowerManager.TargetT3122 – This is the acceptable value of T3122 that the power management loop attempts to achieve. If the actual value of T3122 is larger than this, the BTS will reduce

[1]The term "process" here does not refer to an operating system process. Here, a "process" is simple an on-going activity inside the OpenBTS software.

its output power. If the actual value of T3122 is small than this, the BTS will increase its output power (if it is not already maximized). It is critical that this target value be within the bounds set by GSM.Timer.T3122Max and GSM.Timer.T3122Min, as described in Section 5.1.

- GSM.PowerManager.Period – This is the adaptation time constant in milliseconds.
- GSM.PowerManger.MaxAttenDB – The maximum allowed attenuation, in dB relative to full power, which determines the minimum output level. This is also the initial attenuation level.
- GSM.PowerManager.MinAttenDB – The minimum allowed attenuation, in dB relative to full scale, which determines the maximum output level. This value is normally zero, allowing the BTS to operate at the maximum power level supported by the hardware.

To disable the automatic power control feature, set the minimum and maximum attenuation levels (MaxAttenDB and MinAttenDB) to the same value, usually zero for maximum power at all times. The CLI "power" command, described in Section 5.5.12, can be used monitor this mechanism or to control upper and lower power bounds as a pair.

5.3 Uplink Power and Timing Control

5.3.1 Uplink Power Control

GSM uses a closed-loop uplink power control, described in GSM 05.08 Sections 4.1-4.2 and GSM 05.05 Section 4.1.1. The available maximum power levels of GSM MSs are given in Table 5.1. A multi-band MS can (and typically will) have different power classes in each supported band. The lowest available power output in any band is 5 dBm. The power control range is set with the configuration parameters GSM.MS.Power.Max and GSM.MS.Power.Min, both expressed in dBm. These are normally set to 5 and 39, respectively. These are global settings, applied to all MSs equally. For example, the effect of setting GSM.MS.Power.Max to something less than 39 in a GSM900 unit is to remove any range advantage that might be had by MSs power class 2. If an MS receives a power command that falls outside of its available power range, that MS will set its output power to the closest level available, maximum or minimum. So there is no risk in setting these bounds wider than what the MS can actually support. It may be desirable, though, in some installations, to limit MS power to prevent interference to other cell sites in the area.

Table 5.1: Maximum output power levels for GSM MSs. From GSM 05.05 Section 4.1.1.

Power Class	**GSM850 GSM900 Max. Ouput**	**DCS1800 Max. Output**	**PCS1900 Max. Output**
1	N/A	1 W (30 dBm)	1 W (30 dBm)
2	8 W (39 dBm)	0.25 W (24 dBm)	0.25 W (30 dBm)
3	5 W (33 dBm)	4 W (36 dBm)	2 W (33 dBm)
4	2 W (33 dBm)	N/A	N/A
5	0.8 W (29 dBm)	N/A	N/A

5.3.2 Uplink Timing Control

GSM uses closed-loop timing advance control, described in GSM 05.10 Section 6. The configuration parameter GSM.MS.TA.Max sets a limit on MS timing advance and can be used to deliberately limit the range of service. The value is expressed in symbol periods of round-trip delay, at about 550 meters per step. The normal value of this parameter is 63, which is also the maximum allowed value and corresponds to a maximum range of 35 km.

5.4 Logging

OpenBTS P2.8 logs to syslogd as facility local7. OpenBTS defines the syslogd logging levels to mean the following:

- EMERGENCY – serious fault associated with service failure or hardware damage
- ALERT – likely service disruption caused by misconfiguration, poor connectivity or some other problem not internal to the software
- CRITICAL – anomalous event that is likely to degrade service
- ERROR – an internal error of the software that may result in degradation of service in unusual circumstances
- WARNING – an anomalous event that may indicate a degradation of normal service
- NOTICE – anomalous event that probably does not affect service but may be of interest to network operators
- INFO – a normal event
- DEBUG – detailed information about internal data structures

Syslogd offers a range of powerful archival, reporting and notification mechanisms, including e-mail notifications and remote logging. The reader is referred to syslogd documentation for further information on the features and configuration of that system.

The overall logging level for OpenBTS is set in the configuration variable Log.Level. Logging levels can be set for a individual source file by defining a new configuration variable of the form "Log.Level.*filname*" with a value equal to the desired logging level. For example, "Log.Level.CallControl.cpp INFO" sets the logging level to INFO for all functions in the file CallControl.cpp. These log levels are dynamic and can also be set and changed in real time with the "config" command (Section 5.5.6).

Some useful logging settings are:

- config Log.Level GSML2LAPDm.cpp INFO – for an L2 trace
- config Log.Level RadioResource.cpp INFO – for an L3 RR trace
- config Log.Level MobilityManagement.cpp INFO – for an L3 MM trace
- config Log.Level CallControl.cpp INFO – for an L3 CC trace

- config Log.Level.SIPInterface.cpp INFO – for a trace of all SIP messages
- config Log.Level.SIPEngine.cpp INFO – for a trace of SIP state machine activity
- config Log.Level.SMSControl.cpp INFO – for a trace of L3 SMS activity

The logging destination is controlled by the configuration of syslogd or rsyslogd, depending on the host system's Unix installation. In most configurations used by Range Networks, the logging mechanism is rsyslogd, configured with /etc/rsyslog.d/OpenBTS.conf. The standard configuration is:

```
local7.*                        /var/log/OpenBTS.log
```

which records all OpenBTS log messages is /var/log/OpenBTS.log. The rsyslogd mechanism offers many other controls and options, including email notifications and routing of log messages to remote sites for network monitoring. See the rsyslogd Unix manual pages for more information on these features.

Log events at the CRIT, ALERT and EMERGENCY levels are treated as special cases inside OpenBTS:

- High level log events are echoed to the OpenBTS console, regardless of the Log.LogFile and Log.Level settings or the configuration of syslogd.
- High level log events are stored in an internal table accessible from the CLI (Section 5.5.2). The maximum size of this table is set with the Log.Alarms.Max configuration value.

5.5 The Command Line Interface (CLI)

The OpenBTS console is also called the "command line interface", or CLI. The CLI allows you to monitor system status and change many operating parameters in real time. From the CLI, use the "help" command to get a list of available commands. Use "help" followed by a command name to get a description of a specific command.

In embedded configurations from Range, OpenBTS runs inside a loop called "runloop.sh" that will automatically restart the application if it crashes or exits, so you can restart OpenBTS from the CLI with the "exit" command. The restart time is about 10 seconds.

5.5.1 Attaching to the OpenBTS CLI

During boot-up, the Linux init process starts OpenBTS from /etc/rc.local using the "screen" utility. The screen utility allows one or more users to access the OpenBTS console with the command "sudo screen -x OpenBTS". There is a single console, so if multiple users are attached, they will see each other typing. To detach from the screen session, use control-A followed by the "d" key. *Do not hit control-C, since that will kill the screen session itself and OpenBTS.*

5.5.2 "alarms" Command

List recent alarms. The number of alarms saved in the list is set by the "Log.Alarms.Max" configuration value.

5.5.3 "calls" Command

List in-progress Q.931 and SMS transactions from the internal transaction table. Displayed information includes:

- transaction id – The key for the corresponding entry in the transaction table that is currently making use of this channel.
- SIP call state
- Q.931/GSM call state
- time since last state change
- subscriber IMSI
- called or calling party number

5.5.4 "cellid" Command

Display or change cell identity parameters. These parameters are:

- MCC – Mobile Country Code (3 digits)
- MNC – Mobile Network Code (2 or 3 digits)
- LAC – Location Area Code (16 bits, 1-65520 are valid values)
- CI – Cell Identity (16 bits, 0-65535 are valid values)

With no arguments, this command displays the current MCC, MNC, LAC and CI values. With arguments

```
cellid <MCC> <MNC> <LAC> <CI>
```

this command sets the parameters to the given values and updates the corresponding GSM.Indentity.* configuration table parameters, as described in Section 4.2. Using the command with arguments will also cause the TMSI Table to be cleared.

5.5.5 "chans" Command

This command displays physical channel status from the channel table (Section 4.4) for active dedicated channels. There are no arguments. The reported values are:

- TN – Timeslot number.
- chan type – The dedicated channel type.
- transaction id – The key for the corresponding entry in the transaction table that is currently making use of this channel.

- UPFER pct – Uplink frame erasure rate, as a percentage.
- RSSI dB – Uplink RSSI at the BTS, in dB with respect to full scale.
- TXPWR dBm – Current uplink transmitter power (from the MS) in dBm.
- TXTA sym – Timing advance in symbol periods.
- DNLEV dBm – Downlink RSSI in dBm as measured by the MS.
- DNBER pct – Downlink bit error rate, as a percentage.

5.5.6 "config" & "unconfig" Commands

This commands display and modify parameters in the configuration table (Section 4.2). The "config" command can be used to inspect, create or modify a configuration table value.

```
config <pattern>
```

lists all configuration parameters that contain given pattern.

```
config <key> <value>
```

Creates or sets the given key-value pair in the configuration table.

```
unconfig <key>
```

removes the associated key-value pair from the configuration table.

For example:

```
OpenBTS> config Example.Value 5
defined new config Example.Value as "5"
OpenBTS> config Example.Value
Example.Value: 5
OpenBTS> unconfig Example.Value
"Example.Value" removed from the configuration table
OpenBTS> config ExampleValue
nothing matching "ExampleValue"
OpenBTS>
```

The config command is possibly the most useful and powerful command in the interface. See Section 4.2 for more information on specific configuration values and their effects.

5.5.7 "endcall" Command

Force the termination of a call or other transaction.

```
endcall <transactionID>
```

5.5.8 "exit" Command

The "exit" command shuts down the OpenBTS and transceiver processes. In embedded applications, OpenBTS is running in a restart loop, so the effect of this command is to restart the OpenBTS GSM stack and its associated transceiver. This process takes about 20 seconds.

The "exit" command with no arguments exits the OpenBTS process immediately. If an argument is given, in seconds, the command will wait up to the given number of seconds for in-progress calls and transactions to clear before exiting. During this wait time, no new calls or transactions will be allowed to start.

5.5.9 "load" Command

Give the current BTS load, in terms of active channels and queue lengths.

```
load
```

The results mean:

- SDCCH load – The number of active SDCCHs out of the total available.
- TCH/F load – The number of active TCH/Fs out of the total available.
- AGCH/PCH load – The number of queued messages awaiting transmission on the AGCH and PCH.
- Paging table size – The number of MSs currently being paged.
- Transactions/TMSIs – The number of active transactions in the BTS and the size of the TMSI table.
- T3122 – The current value of the T3122 hold-off timer, in seconds. See Section 5.2 for details.

5.5.10 "notices" Command

Print the copyright and legal notices associated with this installation of OpenBTS.

5.5.11 "page" Command

Page a given IMSI. Since there is no real transaction associated with this page, the MS will be rejected when it attempts to establish a dedicated channel to the BTS. This command is provided for testing purposes.

```
page <IMSI>
```

5.5.12 "power" Command

Inspect or change the downlink power parameters described in Section 5.2. With no arguments, this command displays the current power setting and bounds. With arguments,

```
power <minAtten> <maxAtten>
```

this command changes the power control bounds.

5.5.13 "sendsms" and "sendsimple" Commands

Either of these commands sends a text message via SMS to a given MS , addressed by IMSI and appearing to originate from a given source address:

```
sendsms <IMSI> <sourceAddress>
sendsimple <IMSI> <sourceAddress>
```

You will then be prompted to enter the message text.

The difference between these is that sendsms operates directly in the SMS control layers of OpenBTS while sendsimple operates by sending an RFC-3428 SIP MESSAGE packet to the OpenBTS SIP port.

5.5.14 "testcall" Command

This command is included in the CLI for development purposes, but not supported by Range.

5.5.15 "tmsis" Command

This command displays or clears the TMSI table (Section 4.3).

```
tmsis
```

prints the current TMSI table.

```
tmsis clear
```

clears the TMSI table.

5.5.16 "version" Command

Print information on the installed version of OpenBTS.

5.5.17 CLI Shell Escape

Any line issued to the CLI starting with "!" is processed as shell command (in "sh"). This feature can be used to execute other applications from inside OpenBTS when only one interface screen is available. Examples follow:

- To see the 10 most recent registration attempts, assuming Log.Level.SIPEngine.cpp is set to INFO or lower,

  ```
  OpenBTS> ! grep Register /var/log/OpenBTS.log | grep IMSI | tail -n 10
  ```

- To access the local Asterisk console,

```
OpenBTS> ! sudo asterisk -r
```

If you then exit the Asterisk shell with "quit", you will return to the OpenBTS CLI.

5.6 Open Registration

Open registration is a mode where all MSs are accepted for registration, regardless of their authentication or provisioning status. Depending on the configuration of Asterisk (Chapter 6 and smqueue (Chapter 7) these unprovisioned MSs may be able to make telephone calls and send text messages. To enable open registration, set Control.LUR.OpenRegistration to regular expression matching the IMSIs to be accepted.

Chapter 6

Asterisk and the Subscriber Registry

One of the distinctive features of OpenBTS is to use a generic VoIP switch to replace the GSM mobile switching center (MSC). In the standard OpenBTS P2.8 deployment, the default VoIP switch is Asterisk 1.8.

The key concept in understanding OpenBTS-SIP integration is that each GSM MS in communication with the BTS unit appears to the VoIP network as a SIP endpoint with the username "IMSIxxxxxxxxxxxxxxx", where xxxxxxxxxxxxxxx is the 14- or 15-digit IMSI from the MS's SIM. The IP address of the SIP user is the IP address of its service BTS. OpenBTS itself is invisible to the VoIP network. It is simply a conduit for the MSs.

6.1 Real Time Asterisk & the Subscriber Registry

Commercial configurations of Asterisk use a so-called "real time" Asterisk configuration, where Asterisk depends on an external sqlite3 database for its SIP registry and parts of its dialplan. In OpenBTS, this registry database is part of an application called the subscriber registry. The subscriber registry database is a standard Asterisk SIP registry, following the standard `sip_buddies` format. This database is located on the same physical computer as the Asterisk server that uses it. The path to this database is given in the SubscriberRegistry.db configuration parameter. The default value is

```
/var/lib/asterisk/sqlite3dir/sqlite3.db
```

Asterisk also expects to find the database file in this location, so for most applications that value should not be changed. Because Asterisk and the subscriber registry access the sqlite3 database through a file interface, they must be running on the same physical server.

6.1.1 Configuring the Subscriber Registry

The subscriber registry is configured using the same sqlite3 database table as OpenBTS, described in Section 4.2.[1] Parameters specific to the subscriber registry are in the SubscriberRegistry.* group.

[1] OpenBTS and the subscriber registry share a configuration table to insure proper coordination of common parameters in configurations where these applications are running on the same physical computer.

6.2 Provisioning New Subscribers

"Provisioning" is the process of creating new subscriber accounts. OpenBTS P2.8subscribers are provisioned like any other SIP subscribers in an Asterisk system, with the following constraints:

- The SIP user name is always "IMSI" followed by the digits of the IMSI.

6.2.1 Using Pre-existing SIMs

OpenBTS systems can use pre-existing SIMs.

Manually

The key to manual provisioning is to determine the IMSI of the SIM used in the MS. Two possible ways to do this are:

1. Get a commercial SIM reader, remove the SIM from the phone and read it. SIM readers are available from Range Networks, part #8425-xxx series, including software automate the provisioning process.
2. Enable Control.LUR.FailedRegistration.Message feature to deliver a "welcome message" to unprovisiooned MSs, which is automatically appended with the IMSI digits. A typical message might be "To activate service, bring this code to our offce: ". See Section 4.2 for more infomation.

Once the IMSI is known, the operator can generate an entry in the subscriber registry and assign the subscriber a phone number in the OpenBTS network. Because there is no roaming relationship, the number assigned to the SIM in the OpenBTS network is independent of the number assigned in any other cellular network, although it may be convenient for the two numbers to be the same. Also, it is possible to provision a handset with no telephone number at all, in which case the provisioned MS cannot accept inbound calls but can still place outbound calls.

Interactive Via SMS

Smqueue and the subscriber registry can be used together to provide an interactive autoprovisioning system based on SMS. The configuration parameters are:

- In smqueue:
  - The SC.Register.* parameter set. See Section 7.4.2
- In OpenBTS:
  - Control.LUR.OpenRegistration must be non-null and the Control.LUR.OpenRegistration.* parameters must be defined. See Section 5.6.

The autoprovisioning process is:

1. The MS attempts a location updating request. Even though the MS is not provisioned, the network will accept the request.
2. While the MS still has an open dedicated channel to OpenBTS, OpenBTS sends it the open registration welcome message, defined in Control.LUR.OpenRegistration.Message. This message is usually something like, "Please respond to this message with your telephone number to receive service." The return address for this message, the OpenBTS configuration parameter Control.LUR.OpenRegistration.ShortCode must match the address of the "register" short code function, defined in the smqueue configuration parameter SC.Register.Code.
3. The user responds to the text message with a telephone number.
4. The SMS response is transferred from the MS to OpenBTS to smqueue where it is delivered to the "register" short code function.
5. The "register" short code function updates the subscriber registry to provision the new user.
6. The "register" short code function generates an SMS confirmation (or error) message to the user, delivered by smqueue to OpenBTS to the MS.

6.3 Emergency Calls

In GSM, the emergency call is a special transaction, distinct from ordinary mobile-originated call setup. OpenBTS P2.8 supports the emergency call transaction and presents the call to a SIP switch at a configurable extension.

For speed and reliability, OpenBTS P2.8 always uses very early assignment (VEA) for emergency call establishment in Um, regardless of the setting of the GSM.VEA configuration parameter.

6.3.1 What the User Dials

There are several standard "emergency numbers" used in different parts of the world: 911, 112, 999, etc. When a mobile phone user enters ones of these dialing codes into an MS, it is not treated as a dialed telephone number. Instead, it is a special code that puts the MS into a special emergency call mode. Most GSM MSs will recognize the dial strings 911, 999 or 112 as an emergency call, regardless of where the MS was sold or where it is being used. When the emergency call is delivered to the BTS, the actual number dialed by the user is not reported by the MS, only the fact that an emergency call has been requested. The routing of an emergency call is configured into the network and has nothing to do with the number actually dialed by the user, as long as the user dials a recognizable emergency number.

When OpenBTS receives an emergency call setup request, it presents the inbound call to its designated emergency call proxy, switch or PBX, which may be different from the proxy, switch or PBX used for normal speech calls. The SIP message headers for the INVITE message are formatted according to 3GPP 24.229 Section 4 and include an encoding of the full cell identity and, optionally, the geocoordinates of the BTS site.

Most call setup operations in OpenBTS are non-queuing and calls are rejected immediately if no channel is available. Emergency calls are an exception to this behavior and may be queued for a few seconds waiting for resources to come available. Furthermore, if an emergency call is placed in a congested cell, OpenBTS

P2.8 will terminate the longest-running non-emergency call in the cell to free a channel for the emergency call.

6.3.2 Configuring OpenBTS to Support Emergency Calls

To support emergency calls in OpenBTS, the following must be configured:

- SIP.Proxy.Emergency – This parameter specifies the IP address and port of the proxy, switch or PBX used for emergency calls.
- Control.Emergency.* – All of the parameters in this group must be defined, with the exception of Control.Emergency.Geolocation, which is optional.
- GSM.RACH.AC – Bit 10 of GSM.RACH.AC must be cleared to indicate that the network supports emergency calls.

See Section 4.2 for details about these parameters. These parameters are all dynamic and can be set or changed at any time without disrupting service.

6.4 Connecting to a VoIP Carrier

In many VoIP installations, the operator will use a commercial VoIP carrier to route calls to and from the PSTN. (In some cases the cellular operator may also own and operate the VoIP carrier.) In this example, we create SIP user corresponding to the VoIP carrier and a dialplan context called "from-trunk" where inbound calls from that VoIP carrier are evaluated and routed to an MS.

First, the SIP user representing the VoIP carrier:

```
[my-US-voip-carrier]
context=from-trunk
type=friend
host=my-US-voip-carrier.com
username=myVoIPCarrierAccountUsername
secret=myVoIPCarrierAccountPassword
canreinvite=no
nat=no
insecure=port,invite
qualify=5000
dtmfmode=auto
disallow=all
allow=ulaw
```

Most of these parameters are provided by the carrier. The one to note is the "context" parameter, which we are defining as "from-trunk". The meaning of this is that inbound calls from the VoIP carrier will be evaluated for routing in the from-trunk context of the dialplan.

Here the dialplan entry from extensions.conf:

```
[from-trunk]
; route incoming calls from the PSTN
exten => s,1,Answer
exten => 17075556025,1,Dial(SIP/IMSI460023159705716)
```

The meaning of this is that inbound calls to 17075556025 are connected to SIP user IMSI460023159705716

6.5 Hybrid GSM/SIP Transactions

6.5.1 Registration ("Location Updating")

When an MS enters a new "location area" in a GSM network, it performs a "location update request" (LUR). The network can also instruct the MS to perform the LUR periodically on a timer. The LUR operation is the GSM analog to a SIP REGISTER, and OpenBTS maps the LUR to a SIP REGISTER as shown in Figure 6.1.

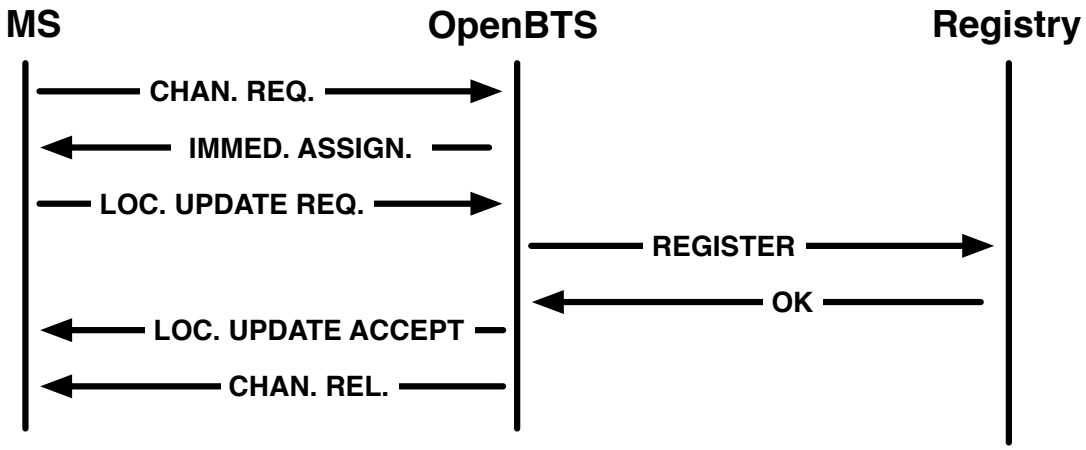

Figure 6.1: GSM location update mapped to a SIP REGISTER (non-authenticating case).

6.5.2 Call Control

Figures 6.2 and 6.3 show the mobile-originated and mobile-terminated call setup cases, using very early assignment for simplicity. In both cases, once the channel is established, the transaction ladder is essentially that of a SIP-ISDN gateway.

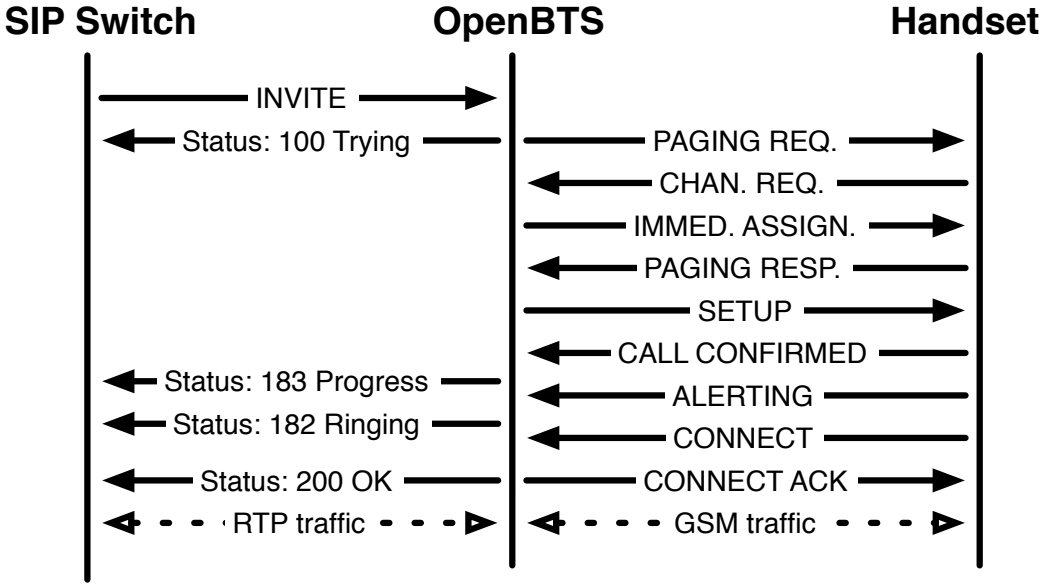

Figure 6.2: A GSM-SIP mobile-terminated call, VEA, normal case.

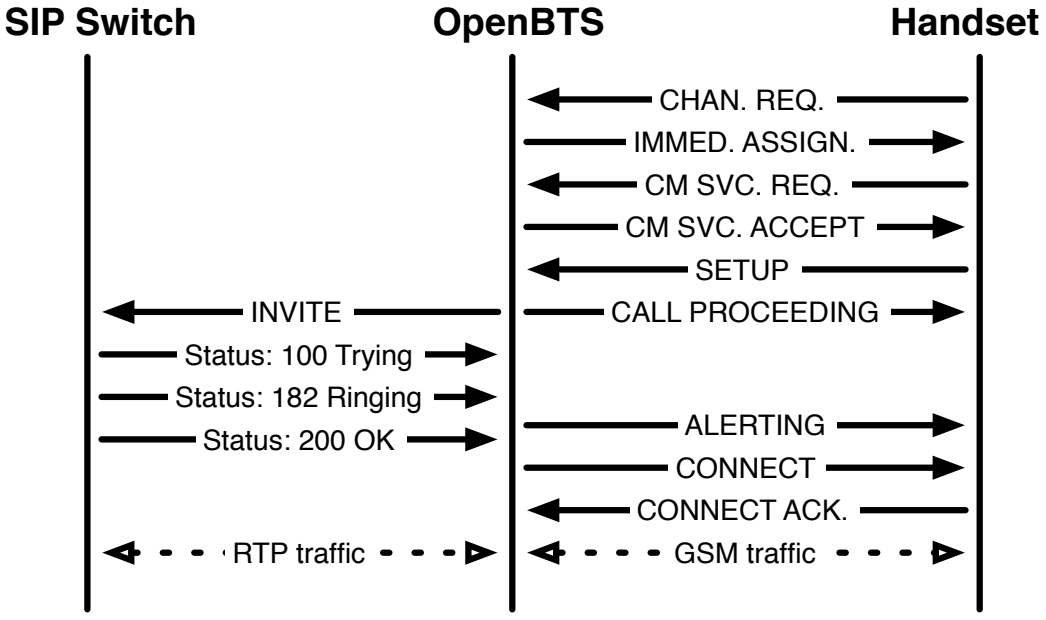

Figure 6.3: A GSM-SIP mobile-originated call, VEA, normal case.

Chapter 7

SMS Text Messaging

GSM text messaging ("short message service" or SMS) is a service akin to e-mail. Users can send and receive 140-byte messages, allowing up to 160 characters using the SMS 7-bit alphabet. Addresses can be ISDN private network, E.164 or e-mail. SMS is a store-and-forward medium and can be held for minutes, hours or even days if the receiving party is not available. Text messaging also uses reliable channels, like the SDCCH, with frame retransmission and acknowledgement in L2, making it tolerant of frame erasure rates in excess of 50%. These properties make SMS a usable medium over much larger coverage areas than speech, in areas with coverage or spotty or weak and where speech quality would be too poor and calls would disconnect too frequently to be useful.

7.1 Internet Messaging Protocols

For OpenBTS to handle SMS in a manner consistent with its design goals, the GSM SMS protocol must be translated to and from some open protocol from the internet world. There are many such protocols, but few are well-suited to SMS.

7.1.1 The "Session" Problem

Most messaging protocols in the IETF/IP world, like XMPP, are built around the notion of a "connected session", a virtual circuit similar the virtual connection of RTP or TCP/IP. This model assumes an "always-on" network connection where the maintenance of the circuit, with occasional keep-alive messages, is cheap and reasonable.

GSM SMS is different. Maintaining the channel is expensive. There is no keep-alive message mechanism. The circuit switched connection is created and destroyed with every transfer in a process that can take hundreds of milliseconds. Each message transfer is an independent transaction. There is no natural notion of a session.

7.1.2 RFC-3428

RFC-3428 is an IETF standard for the transfer of short messages over the internet. Among IETF/IP protocols for messaging, RFC-3428 is special in that it supports "page mode" messaging, without any

notion of a session; there is no INVITE to start the transaction or BYE to end it. This makes it a natural fit for SMS. RFC-3428 is straightforward. The sending entity sends a SIP MESSAGE method to the intended receiver. The receiver gives one of the standard SIP responses, preferably 200 OK or 202 Queued to indicate a successful transfer, or a 4xx or 5xx response to indicate failure.

7.2 Smqueue

The delivery of each text message depends on a store-and-forward facility in the network. Smqueue provides this facility. In the P2.8 release, smqueue is provided with source code under GPLv3.[1]

7.2.1 Design and Operation of Smqueue

The core of smqueue is a queue of messages awaiting delivery. Messages wait in this queue, potentially through multiple delivery attempts, until delivery is confirmed or until the message is determined to be undeliverable. The operation is similar to that of an email server.

7.2.2 Addressing in Smqueue

Smqueue recognizes two kinds of address: ISDN/E.164 numeric addresses and SIP usernames. Any all-numeric address is assumed to be an ISDN/E.164 address and smqueue will attempt to resolve it to a SIP username using the subscriber registry. Any address that is not all-numeric is assumed to be a SIP username in the operator's network.

7.2.3 Configuration of Smqueue

Smqueue is configured through an sqlite3 database table using the same schema as the OpenBTS configuration table. As of release P2.8, the configuration can be changed only by restarting smqueue after changing the database. Some configuration parameters of note are:

- Log.Level, Log.Level.* – These are logging controls that behave the same as those in OpenBTS.config. See Section 5.4 for more information.
- SIP.myIP – The IP address of the smqueue machine as seem by the subscriber registry server.
- SIP.myIP2 – The IP address of the smqueue machine as seen by remote gateways.
- SIP.global_relay – The IP address of a remote RFC-3428 server for delivery of non-local messages. If no such gateway is available, this parameter should be an empty string (''). This parameter can be used to build a hierarchy of smqueue servers for large networks.
- BounceMessage.* – A set of error messages sent back to MSs when submitted messages are undeliverable.
- SC.* – Configuration parameters for specific short code functions, not for smqueue itself. See Section 7.4.

[1] This will not be true in future releases.

7.3 Text Messaging in GSM

GSM 04.11 and 03.40 define conventional SMS in five layers:

1. L1 is taken from the Dm channel type used, either SDCCH or SACCH. This layer terminates in the BSC.
2. L2 is normally LAPDm. This layer terminates in the BTS.
3. L3, the connection layer, is defined in GSM 04.11 Section 5. This layer terminates in the MSC.
4. L4, the relay layer, is defined in GSM 04.11 Section 6. This layer terminates in the MSC.
5. L5, the transfer layer, is defined in GSM 03.40. This layer terminates in the SMSC.

As a general rule, every message transferred in L(n) requires both a transfer and an acknowledgment on L(n-1).

In the OpenBTS realization of SMS, there is no MSC, so L3 terminates in OpenBTS and L4 is a SIP relay to smqueue, which takes the place of the SMSC.

7.3.1 Layers of SMS in OpenBTS and Smqueue

We will now consider the handling of each layer of SMS by OpenBTS and smqueue.

SMS in L3

The Um L3 part of SMS uses three messages:

- CP-DATA to transfer an RPDU across Um and into L4.
- CP-ACK to acknowledge the transfer of an RPDU across Um and into L4.
- CP-ERROR to report the failure to transfer an RPDU to L4.

An RPDU is a "relay (layer) protocol data unit", which is just an encapsulation of a message from L4. The operation in L3 is simple. The entity that needs to transfer an RPDU sends it in a CP-DATA message. The receiving entity responds with CP-ACK or CP-ERROR. Transactions are non-overlapping.

The action of OpenBTS upon receiving CP-DATA from an MS is to verify the correct encoding of the L3 part of the message and respond with CP-ACK or CP-ERROR. OpenBTS then extracts the RPDU, transfers it to smqueue as an application/vnd.3gpp.sms MIME payload in a SIP MESSAGE method and waits for a response (200 OK or 202 Queued for success or 4xx or timeout for failure).

After sending CP-DATA to an MS, OpenBTS waits for CP-ACK or CP-ERROR, proceeding after CP-ACK or aborting the transaction after CP-ERROR.

SMS in L4

The Um L4 part of SMS uses four messages:

- RP-DATA to transfer a TPDU across Um and into L5.
- RP-ACK to acknowledge the transfer of a TPDU across Um and into L5.
- RP-ERROR to report the failure to transfer an TPDU to L5.
- RP-SMMA for the MS to report that it has more memory available to receive SMS messages (not supported by OpenBTS P2.8).

An TPDU is a "transfer (layer) protocol data unit", which is just an encapsulation of a message from L5. OpenBTS translates between SIP and SMS L4 as follows:

- RP-DATA – MESSAGE method with the RPDU as an application/vnd.3gpp.sms MIME payload
- RP-ACK – 200 OK or 202 Queued response
- RP-ERROR – any other response or timeout
- RP-SMMA – not supported in P2.8

SMS in L5

The Um L5 part of SMS uses these message:

- SMS-SUBMIT to transfer a text message from the MS to the network.
- SMS-DELIVER to transfer a text message from the network to the MS.

OpenBTS transfers L5 PDUs (TPDUs) as opaque payloads. Smqueue manipulates L5 headers as needed to convert SMS-SUMBIT TPDUs into SMS-DELIVER TPDUs during the delivery process.

7.3.2 RFC-3428/SMS Transaction Ladders

Now we take a look at all of the GSM layers and the SIP transactions together.[2]

Mobile Terminated SMS

Figure 7.1 shows a complete mobile terminated SMS transfer, where the network, through smqueue, transfers a text message to the MS. The message arrives at smqueue from the outside world addressed either to a SIP user or to a numeric address. Smqueue resolves the destination address to an IMSI-based SIP user

[2]Strictly speaking, page-mode message transfers are not transactions in SIP, since they are not contained within an INVITE-BYE session. However, these transfers *are* transactions inside of OpenBTS and will be referred to as transactions throughout the OpenBTS documentation.

name and forwards the message to OpenBTS. OpenBTS pages the MS, establishes a channel, transfers the message as SMS and then responds to smqueue with 200 OK.

The most common failure in the mobile terminated transfer is that the MS does not respond to paging. In this case, smqueue never receives any response. The message remains in the smqueue delivery queue and another delivery attempt will be made in a few minutes.

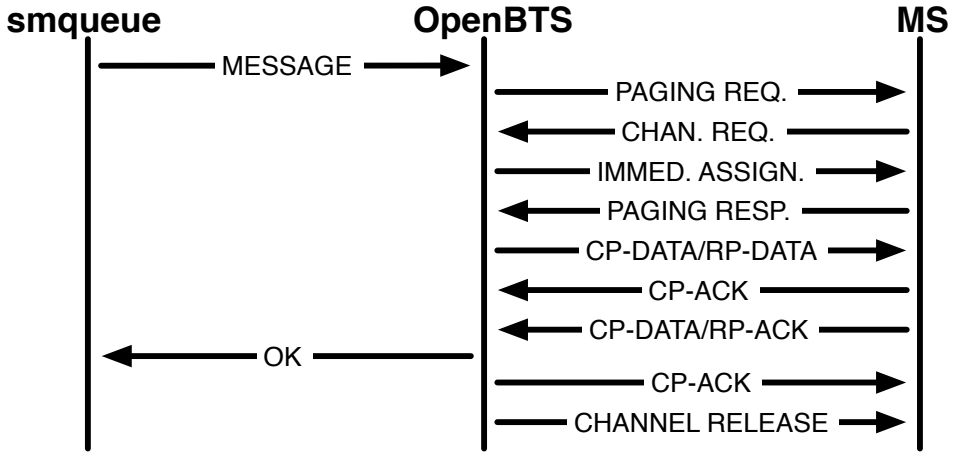

Figure 7.1: Mobile-terminated SMS transfer with no parallel call, normal case.

Mobile Originated SMS

Figure 7.2 shows a complete mobile originated SMS transfer, where the MS, transfers a text message to smqueue for later delivery to its addressee. The message originates in the MS with a numeric address. The MS establishes a radio channel to OpenBTS and then sends the text message TPDU in an RP-DATA message. OpenBTS translates the TPDU to a SIP MESSAGE method and sends that to smqueue. Smqueue responds with OK and then OpenBTS responds to the MS with RP-ACK.

7.4 Short Code Applications

Short codes are local addresses within smqueue that terminate in local application code. A message sent to a short code becomes an input argument to a short code handler function, instead of being delivered to another user. Short code functions provide a means of writing interactive applications based on text messaging. A typical SMS-based application would normal comprise several short code addresses and handlers sharing common data.

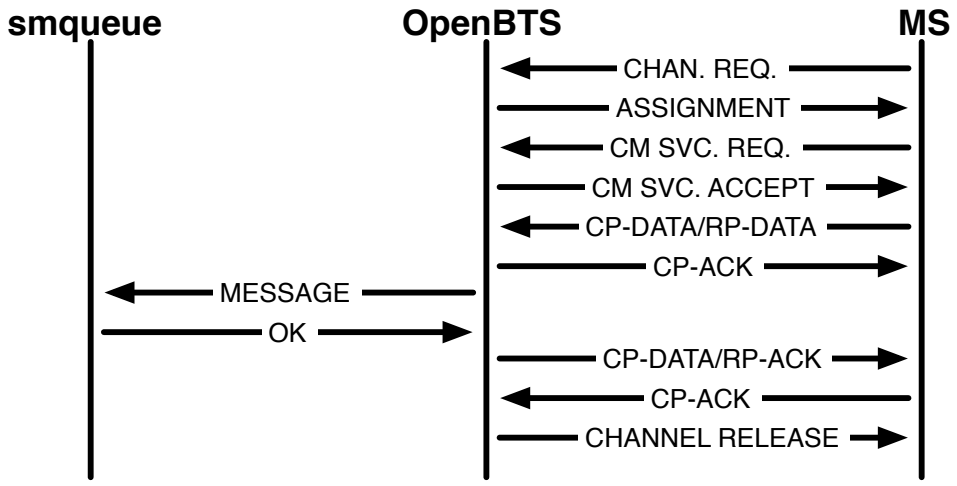

Figure 7.2: Mobile-originated SMS transfer with no parallel call, normal case.

7.4.1 Short Code Implementation

The short code implementation in OpenBTS P2.8 is primitive but functional. Each short code handler is a C++ function coded directly into smqueue/smcommands.cpp.[3] The arguments to a short code handler are the source IMSI of the message, the message text and a short_code_params data structure into which any reply message can be written. The return value from a short code handler is a status code called short_code_action. See smqueue/smcommands.cpp for examples.

Once a short code handler function is defined, it must also be registered at a numeric address. This happens in SMqueue::init_smcommands(), also defined in smqueue/smcommands.cpp.

7.4.2 Existing Short Code Applications

There are a few interesting short code applications built in to the standard release of smqueue, although they are all simple applications each requiring only a single handler function. Not all short codes are documented here. Undocumented short codes are left disabled in the default configuration. Refer to the smcommands.cpp source code for information on undocumented short codes. To disable any short code function, set its short code address to null in the configuration table.

Autoprovisioning ("Register")

The autoprovisioning application allows OpenBTS users to create new entries in the subscriber registry via SMS. The user sends his desired telephone number in a text message to the autoprovisioning short code address. If the requested number has an acceptable number of digits and is not already assigned to a user, the autoprovisioning handler function will perform the steps described in Section 6.2.1 to provision the new user into the subscriber registry. The autoprovisioning short code handler function is called

[3] In future releases, short codes will be implemented as external modules, not directly in smqueue. However, the calling interface will be maintained to allow reuse of legacy short code source code in these nodules.

shortcode_register and it is configured through the SC.Register.* parameters in the smqueue configuration table.

For autoprovisioning to work, OpenBTS must also be configured with open registration enabled so that unprovisioned MSs will show service and be capable of sending text messages. The open registration welcome message can be a powerful way to advertise this feature, especially if the return short code of the welcome message is the same as the short code of the autoprovisioning function. See Section 5.6 for more information.

Email

Any text message send to this address is expected to start with an internet email address. The short code function forwards the message text to that email address using the Unix sendmail utility.

Info

This short code handler generates a brief report of smqueue status, returned in a text message. The implementing function is called shortcode_four_one_one. This short code handler can be configured with the SC.Info.* parameters.

Whiplash Quit

The whiplash_quit shortcode handler shuts down smqueue if the correct passphrase is sent to the correct address. This function is useful for development, but should be disabled in deployed systems. This short code handler can be configured with the SC.WhiplashQuit.* parameters.

Zap Queued

The shortcode_zap_queued handler can delete a message with a specific hash tag or can delete all messages in the queue older than 5000 seconds. This short code application requires a password. This short code handler can be configured with the SC.ZapQueued.* parameters in the smqueue configuration table.

Chapter 8

Other GSM Services

This chapter covers 2G GSM services beyond speech and SMS text messaging.

8.1 Radio Resource Location Protocol (RRLP)

RRLP is the protocol used between the network and MS to manage location services (LCS). All handsets sold in the US market are required, under E911 regulations, to include LCS support. At least 90% of handsets in use in the US and EU today support this protocol.

The GSM specifications (02.71, 03.71, 04.30, 04.31 and others) define several possible techniques for determining the location of an MS, but by far the most commonly used is assisted GPS (AGPS) and nearly every handset that supports RRLP also supports the AGPS mode. GPS is the well-known satellite-based positioning system and AGPS-capable handsets include GPS receivers.

8.1.1 The Need for Assistance

Calculation of a position from GPS measurements requires the following information:

- Pseduoranges. These are timing measurements made on the GPS signal. They are called pseudoranges because they are directly related to the distances from the GPS satellites. Pseudoranges are derived from signal delay measurements called "codephases".
- Ephemerides. These are detailed descriptions of the current satellite orbits, used to calculate the exact location of each satellite in space.
- Ionospheric model. The radio propagation delay through the ionosphere can vary by hundreds or even thousands of nanoseconds and so must be corrected in the positioning calculations to prevent errors of hundreds of meters in the final positioning results.
- Seed position. The GPS positioning signal (the "C/A code") has a period of 1 ms, corresponding to a distance of about 300 km. This means that there are *many* potential solutions for the GPS positioning equations in an irregular lattice around the Earth. If the GPS receiver does not know its true position within about 200 km, it must check all of these potential solutions in a brute force

search before making a reliable position estimate, a process that can take 20 minutes or longer. To avoid this delay, the GPS receiver requires a seed position within 200 km of its true location.

- Rough current time. Current time must be known to within a few seconds to make rough estimates of satellite position and bootstrap the positioning calculations. Like seed position, rough time can determined through a brute-force search, but that is very time-consuming.

In normal GPS operation, the pseudoranges are measured directly from the GPS signal, the ephemerides and ionospheric model are encoded into the GPS signal and the seed position is taken from the most recent successful positioning attempt (usually saved in non-volatile memory between power-cycles). Often, though, the GPS receiver in a cellular handset has no useful seed position saved and receives the GPS signal too poorly to decode the critical data that is encoded in them. However, even under these conditions, the GPS receiver can often measure pseudoranges and "could" estimate a position if the other information were provided to it through some other channel. The process of providing this other information is called "assistance" or "aiding", hence the terms "assisted GPS" and "aiding information".

The RRLP specification defines two basic modes of operation:

1. MS-based, where the network provides ephemeris and ionospheric parameters, rough time and seed position and the GPS receiver in the MS performs the positioning calculation and
2. network-based, where network provides rough codephases to the MS to bootstrap the measurement process, the MS makes actual codephase measurements, the MS reports the raw codephases back to the network and the network performs the positioning calculation.

OpenBTS P2.8 supports only the MS-based RRLP mode.[1]

8.1.2 The RRLP Server

Overall Design

In keeping with the OpenBTS design philosophy, RRLP is not implemented inside OpenBTS, but in an external server that communicates with OpenBTS through an HTTP interface at the L3/L4 boundary. The role of OpenBTS is to transfer L4 RRLP APDUs between the MS and the RRLP server. From OpenBTS to the server, these APDUs are encoded in the URLs of HTTP requests. From the server to OpenBTS, APDUs are passed in HTTP responses. The RRLP server can be run in each BTS unit, in each cell site or from a single central location.

Because the RRLP server is HTTP based, its actual implementation is as a CGI program in a standard web server.

Source of Aiding Information

The primary function of the RRLP server in network-assisted positioning is to provide aiding information. The RRLP server acquires the current GPS almanac and ephemerides for aiding information from publicly available sources on the internet.[2] These parameters total about 2 kB and are downloaded every hour.

[1] Future versions of OpenBTS will support network-based positioning as well.

[2] Future versions of OpenBTS will support direct connection of GPS receivers to the RRLP server.

The AGPS seed location is provided by the BTS unit that submits the HTTP request to the RRLP server. This location is defined in the RRLP.* configuration parameters listed in Section 4.2.

Rough current time comes from the time-of-day clock on the machine running the RRLP server. The machine running the RRLP server should also be running NTP (network time protocol) to calibrate its time-of-day clock.

8.1.3 RRLP Service Controls and Configuration

Server Configuration

All parameters the control the behavior of the RRLP server are given to it in the URL of the service request. This means that the server itself has no configuration; all of the configuration is in the client side in the OpenBTS configuration table. See the GSM.RRLP.* configuration parameter set in Section 4.2 for more information.

Invoking RRLP

The following parameters control whether or not all MS are queried for RRLP position information:

- Control.LUR.RRLPQuery
- Emergency.RRLP

See Section 4.2 for more information.

Getting RRLP Results

RRLP results are written to the subscriber registry database, to a table called "RRLP". The schema is:

```
CREATE TABLE IF NOT EXISTS RRLP (
  id INTEGER PRIMARY KEY, -- integer key
  name VARCHAR(80) not null, -- SIP user name (IMSI)
  latitude real not null, -- WGS84 latitude in degrees
  longitude real not null, -- WGS84 longitude in degrees
  error real not null,  -- circular error estimate in meters
  time text not null -- Unix time of fix
)
```

Document History

Date	Doc. Rev.	Changes
19 Sept 2011	1	derived P2.8 manual rev 1 fron C2.8 manual rev 3